UNIVERSITÄT DORTMUND
FACHBEREICH INFORMATIK

DIPLOMARBEIT zum Thema

Auswertung funktioneller Perfusionsuntersuchungen der Nieren bei gleichzeitiger Unterdrückung der Atembewegung des Organs

Jens Martin

I N T E R N E B E R I C H T E
I N T E R N A L R E P O R T S

Betreuer:
Prof. Dr. Bernd Reusch
PD Dr. med. Dipl.-Phys. Thomas Hackländer

Lehrstuhl Informatik I
Otto-Hahn-Strasse 16
44227 Dortmund

Vorwort

Im Mittelpunkt der vorliegenden Diplomarbeit steht ein System zur Unterstützung der medizinischen Nierenfunktionsdiagnostik, basierend auf Aufnahmesequenzen der Magnetresonanztomographie.

Die Idee zu dieser Diplomarbeit stammt von PD Dr. med. Dipl.-Phys. Thomas Hackländer vom Klinikum Wuppertal GmbH, mit dem ich schon im Rahmen der Projektgruppe ANOMALIA zusammengearbeitet habe. Während sich ANOMALIA mit der Erkennung von pathologischen Strukturen in statischen MRT-Bildern beschäftigte, kann das hier vorgestellte System ein dynamisches Kontrastmittelverhalten in zeitlich aufgelösten Sequenzen untersuchen.

Da die herkömmliche manuelle Bearbeitung der Nierensequenzaufnahmen ein sehr zeitaufwendiger Prozeß ist, bestand der Wunsch der Radiologen nach einer computerunterstützten, automatischen Auswertung. Hieraus wurden Thema und Problemstellung dieser Arbeit abgeleitet.

An dieser Stelle möchte ich mich bei meinen Betreuern Prof. Dr. Bernd Reusch und PD Dr. med. Dipl.-Phys. Thomas Hackländer für ihre freundliche Unterstützung bedanken. Mit ihrer Geduld und Zuversicht haben sie mich erfolgreich ans Ziel geführt. PD Dr. med. Dipl.-Phys. Thomas Hackländer stand mir in zahlreichen Gesprächen zur Wissensakquisition und zur Diskussion der Programmimplementierung hilfreich zur Seite.

Herrn Dr. Jörg Stattaus vom Klinikum Wuppertal sei ebenfalls Dank ausgesprochen für seine freundliche Unterstützung. Seine unermüdlichen Praxistests von Vorstadien des fertigen Programms, Diskussionen zur Benutzeroberfläche, sowie zahlreichen Literaturhinweise haben mir sehr geholfen. Auch für seine Hinweise zur Korrektur des schriftlichen Teils der Arbeit bin ich dankbar.

Darüber hinaus möchte ich Dipl.-Inform. Jens Hiltner danken, der mir bei Fragen zum theoretischen Teil der Arbeit stets hilfreich zur Seite gestanden hat, wertvolle Hinweise zur Literatursuche gegeben hat und ebenfalls durch Korrekturen des schriftlichen Teils der Arbeit geholfen hat.

Weiterer Dank geht an Dr. Inform. Dipl.-Ing. Thomas Lehmann für seine freundliche Unterstützung und Hilfestellung im Bereich der Bildregistrierung sowie an Dipl.-Biol. Karsten Hannig für seine konstruktiven Anregungen beim Korrekturlesen der Arbeit.

Meinem Bruder, Dipl.-Des. Tim Martin, sei für seine Unterstützung und seine Ratschläge zur grafischen Gestaltung von Abbildungen und Titelbildern gedankt.

Meiner lieben Annette möchte ich von Herzen für ihre Geduld, Ermutigung und Liebe in den Monaten der Erstellung dieser Arbeit danken. Darüber hinaus hat ihr unermüdliches Gegenlesen viel dazu beigetragen, kleine und große Fehler auszumerzen.

Das im Rahmen dieser Diplomarbeit entwickelte Softwareprodukt wurde zum Abgabezeitpunkt der Diplomarbeit am Lehrstuhl Informatik I sowie am Klinikum Wuppertal, Radiologische Abteilung, hinterlegt.

Dortmund, im Februar 2000

Inhaltsverzeichnis

1 Einleitung ... 5
1.1 Problembeschreibung ... 5
1.2 Aufgabenstellung ... 5
1.3 Aufbau der Arbeit ... 6
2 Anatomie und Pathophysiologie der Niere ... 7
2.1 Anatomie der Niere ... 8
2.2 Pathophysiologie der Niere ... 9
2.3 *Renovaskuläre Hypertonie* ... 11
2.3.1 Blutdruck und Blutdruckregulierung ... 11
2.3.2 Hypertonie ... 13
2.3.3 Nierenarterien*stenose* ... 14
3 Schnittbilddiagnostik ... 17
3.1 Magnetresonanztomographie ... 18
3.2 MRT-Aufnahmetechniken ... 20
3.2.1 Turbo-Flash-Sequenzen ... 22
3.2.2 Bilddarstellung und Fensterung ... 22
4 Funktionale Beschreibung des Systems ... 24
4.1 Bildmaterial ... 25
4.2 Bildverarbeitung mit IMAGEJ ... 25
4.3 Korrektur der Atmungsbewegung ... 26
4.3.1 Kepstrum-Filter ... 27
4.3.2 Algorithmus zur Bewegungskorrektur von Nierenserien ... 28
4.4 Berechnung der Meßkurven ... 29
4.4.1 Differenzierung der Nierengewebe ... 29
4.5 Manuelle Auswertung ... 31
4.6 Export der Ergebnisse ... 32
4.7 Übersicht ... 32
5 Grundlagen der Bilddarstellung und Bildverarbeitung ... 34
5.1 Bilder ... 35
5.1.1 Bildeigenschaften ... 35
5.1.2 Modifikation der Grauwertverteilung ... 36
5.1.3 Differenzbilder ... 37
5.2 Ortsraum ... 38
5.2.1 Lokale Operatoren im Ortsraum ... 38
5.2.2 Lineare Glättungsoperatoren ... 39
5.2.3 Median-Filter ... 39
5.2.4 Lokale Gradientenoperatoren ... 40
5.3 Frequenzraum ... 41
5.3.1 Die (kontinuierliche) Fourier-Transformation ... 42
5.3.2 Wichtige Eigenschaften der kontinuierlichen Fourier-Transformation ... 44
5.3.3 Die diskrete Fourier-Transformation ... 46
5.3.4 Wichtige Eigenschaften der diskreten Fourier-Transformation ... 47
5.3.5 Herleitung der diskreten Fourier-Transformation ... 48
5.4 Interpolation ... 49
5.4.1 Interpretation als Faltung ... 49
5.4.2 Die kubische Interpolation ... 50
6 Kepstrum-Filterung zur Translationskorrektur ... 52
6.1 Korrektur von Translationen ... 53
6.1.1 (Kreuz-)Korrelation ... 53
6.1.2 Diskrete Ähnlichkeitsmaße ... 54
6.1.3 Korrelation im Frequenzraum ... 55
6.2 Kepstrum-Filterung ... 58
6.2.1 Zweidimensionale Kepstrum-Filterung ... 59
6.2.2 Diskrete Berechnung ... 62
7 Verbesserungen der Kepstrum-Filterung ... 64
7.1 Vorverarbeitung ... 65
7.2 Fensterung ... 65
7.3 Zeropadding ... 65
7.4 Histogrammangleich ... 66

7.5 Optimierung durch die Hartley-Transformation 67
7.6 Steigerung der Genauigkeit 68
7.7 Inhomogene Verschiebung der Bildinhalte 68
7.7.1 Nachjustierung 70
7.7.2 Raster-Transformation 71
7.8 Segmentierung 74
8 Differenzierung der Nierengewebe 76
8.1 Berechnung der Meßkurven 77
8.1.1 Gesamtniere 77
8.1.2 Nierenrinde und Nierenmark 78
8.2 Diagnose 80
9 Programmierung 81
9.1 Allgemeine Informationen zur Implementierung 82
9.1.1 Wahl der Programmiersprache 82
9.1.2 Arbeitsmittel und Arbeitsumgebung 82
9.1.3 Quelltextkonventionen 83
9.2 Bildverarbeitung mit IMAGEJ 85
9.2.1 *Plugin* und *PluginFilter* 85
9.2.2 *ImagePlus* und *ImageStack* 86
9.2.3 *ImageWindow* und *ImageCanvas* 87
9.3 Übersicht 88
9.4 Import der Nierensequenzen 89
9.5 Das *Plugin* zur renalen Funktionsanalyse 90
9.5.1 Das Paket *kidney* 91
9.5.2 Das Paket *extension* 93
9.5.3 Das Paket *util* 94
9.6 Kepstrum-Filterung 95
9.6.1 Das Paket *jigl* 95
9.6.2 Das Paket *cepstrum* 96
9.7 Berechnung der SIT-Kurven 99
9.7.1 Das Paket *sit* 99
9.8 Manuelle Bearbeitung 102
9.8.1 Das Paket *kidney* 102
9.8.2 Das Paket *extension* 102
9.8.3 Das Paket *sit* 103
9.9 Ergebnisdarstellung, Justierung und Export 104
9.9.1 Das Paket *sit* 104
9.10 Weitere Pakete 107
9.10.1 Das Paket *segmentation* 108
9.10.2 Das Paket *scaler* 109
10 Installation und Bedienung 110
10.1 Installation 111
10.1.1 Hard- und Softwarevoraussetzungen 111
10.1.2 Installation und Programmstart 111
10.2 Bedienungsanleitung 112
10.2.1 IMAGEJ 112
10.2.2 Import der Nierensequenzen 113
10.2.3 Das *Plugin* zur renalen Funktionsanalyse 115
10.2.4 Durchführung der Kepstrum-Filterung 117
10.2.5 Berechnung der SIT-Kurven 121
10.2.6 Justierung der Klassifizierungsergebnisse 124
10.2.7 Manuelle Bearbeitung von Nierensequenzen 125
10.2.8 Voreinstellungen 127
10.2.9 Sonstige Bedienungselemente 131
11 Ergebnisse 132
Anhang A: Literaturverzeichnis 134
Anhang B: Abbildungs- und Tabellenverzeichnis 139
Anhang C: Index 141

1 Einleitung

1.1 Problembeschreibung

Für viele weitverbreitete Erkrankungen, wie z.B. den krankhaft erhöhten Blutdruck, ist häufig eine Funktionseinschränkung der Nieren verantwortlich. Häufig sind dabei Verengungen oder Verschlüsse der zuleitenden Blutgefäße (Nierenarterienstenosen, NAS) verantwortlich, die dann krankhafte Veränderungen des Nierengewebes verursachen können. Mit einer funktionellen Untersuchung der Nierenperfusion möchte der Mediziner klären, ob und in welchem Umfang Störungen vorliegen. Die Beurteilung der Nierenfunktion erfolgt heute noch weitgehend mit nuklearmedizinischen Methoden. Es zeichnet sich aber ab, daß die Kernspintomographie ohne Strahlenexposition der Patienten ähnliche Ergebnisse liefern kann, wie die Nuklearmedizin.

Beim Einsatz der Kernspintomographie oder auch Magnetresonanztomographie (MRT) als diagnostisches Mittel werden Serienaufnahmen der Nieren über einen Zeitraum von drei Minuten in Zeitintervallen von 2 Sekunden angefertigt. Dabei wird dem Patienten ein Kontrastmittel intravenös verabreicht, um anhand der gemessenen Kontrastmittelpassage durch die Nieren eine quantitative Funktionsbewertung vorzunehmen. Abbildung 1.1 zeigt ein Beispiel einer solchen Serie. Störend wirkt sich dabei die Atmung der Patienten aus, die eine periodische Verschiebung von Organen und Geweben im Bauchraum (Abdomen) während der Bildgewinnung bewirkt. Für einen exakten Befund ist jedoch die Bewertung von gleichen Ausschnitten der Nieren in jedem Bild der Serie von Bedeutung. Die atmungsbedingte Organverschiebung verhindert dies jedoch, so daß eine manuelle Justierung für jedes Bild nötig ist. Da bei der beschriebenen Technik eine Serienaufnahme 90 Bilder umfaßt, ist es für den Radiologen äußerst zeitaufwendig, Meßkurven zu erstellen. Die Automatisierung und Optimierung dieser Bildauswertung ist Thema dieser Diplomarbeit.

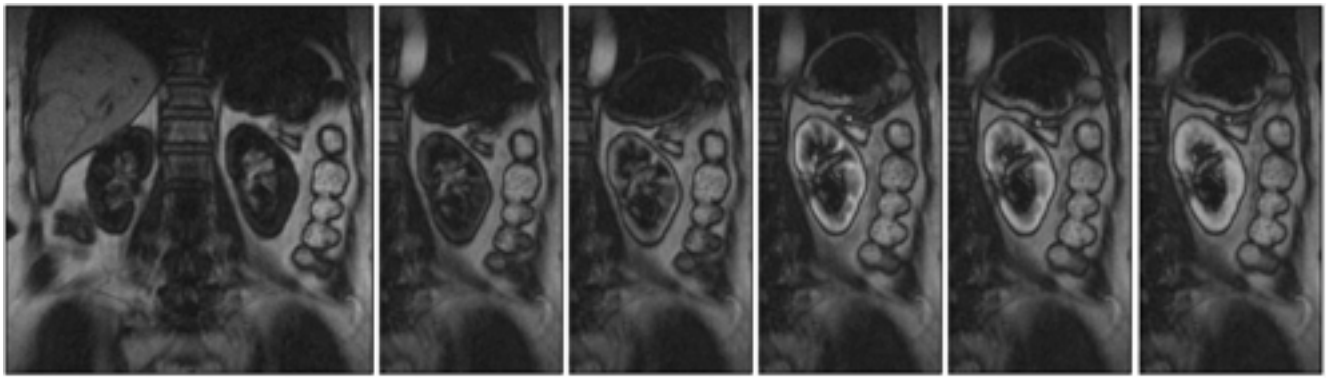


Abbildung 1.1: Beispiel einer MRT-Nieren-Serie. Links ein Vollbild vom Beginn der Aufnahme, danach Halbbilder der linke Niere aus dem weiteren Verlauf.

1.2 Aufgabenstellung

Ziel der Diplomarbeit ist die Entwicklung eines Systems zur automatischen Erstellung von Meßkurven der funktionellen Nierenperfusion. Die Software soll mit der Programmiersprache JAVA erstellt werden, um eine maximale Portabilität zu gewährleisten und um den häufig inhomogenen Rechnernetzen in Krankenhäusern Rechnung zu tragen.

Ausgangspunkt der Überlegungen bei der Programmerstellung sollen dabei die atmungsbedingten Organverschiebungen sein. Sind die Nieren in jedem Bild der Serie an identischer Position zu finden, so müssen nur im ersten Bild der Sequenz die interessanten Regionen (engl. regions of interest, ROI) markiert werden und können auf die restlichen Bilder der Serie projiziert werden. Weiterhin soll eine

möglichst automatische Erstellung der Meßkurven angestrebt werden, welche zur detailierten Auswertung mit anderen Programmen exportiert werden können.

Das System sollte auch in der Lage sein, bei Versagen der automatischen Korrektur der Organverschiebungen – z.B. aufgrund schlechter Aufnahmeparameter oder Aufnahmeeinstellungen – eine manuelle Auswertung zu ermöglichen. Viele Hersteller von Magnetresonanztomographen liefern dazu passende Computer-Arbeitsplätze (engl. workstations) und Programme, mit denen eine manuelle Auswertung i.d.R. möglich ist. Wird diese Funktionalität mit in das Programm dieser Arbeit integriert, entsteht ein kompletter Arbeitsplatz zur Perfusionsuntersuchung.

Die Implementierung soll mindestens bis zum lauffähigen Prototypen durchgeführt werden.

1.3 Aufbau der Arbeit

Mit dieser Arbeit wird ein weiterer Brückenschlag der Informatik in das Gebiet der medizinischen Diagnostik gemacht, siehe insbesondere auch [RFH98] und [HFR99]. Sie soll zeigen, wie weit die Anwendungsmöglichkeiten der computerunterstützte Bildauswertung im Bereich der Radiologie reichen und versteht sich somit auch als Studie des technisch und medizinisch aktuellen Wissensstandes. Gleichzeitig möge sie sowohl Radiologen als auch Informatiker ermutigen, weitere Projekte und Anwendungen interdisziplinär anzugehen. Die medizinische Informatik wird den Radiologen jedoch nie ersetzen können, sie kann lediglich helfen, die Diagnostik zu beschleunigen und zu optimieren.

In der fachübergreifenden Thematik liegt aber gleichzeitig auch das Problem, die Inhalte so darzustellen, daß sie den Lesern beider Fachrichtungen gleichermaßen zugänglich sind. Der Schwerpunkt sollte – und wird – bei der Informatik liegen, jedoch wurden Gliederung und Inhalt der Kapitel etwas anders gewählt, als dies eigentlich bei einer Diplomarbeit der Informatik die Regel ist.

Um ein Verständnis des medizinischen Hintergrunds sowie der angewendeten Untersuchungsmethoden zu vermitteln, wird zunächst in Kapitel 2 auf die Anatomie und Pathophysiologie der Niere eingegangen. Darauf folgt mit Kapitel 3 eine Einleitung in die medizinische Schnittbilddiagnostik. Hier wird die Aufnahmetechnik der in dieser Arbeit untersuchten Turbo-Flash-Sequenzen erläutert. Kapitel 4 beschreibt zunächst die Arbeitsweise des Gesamtsystems. Dieser Überblick soll inbesondere Lesern der medizinischen Fachrichtung helfen, die in den darauffolgenden theoretischen Kapiteln behandelte Thematik und Methodik besser einordnen zu können. In Kapitel 5 werden nun die notwendigen Grundlagen der Bildverarbeitung behandelt, bevor in Kapitel 6 das Kernstück der Diplomarbeit, die Anwendung des Kepstrum-Verfahrens zur Korrektur der Atmungsbewegung, theoretisch und praktisch erklärt wird. Kapitel 7 faßt einige Verbesserungsmöglichkeiten der Kepstrum-Filterung zusammen und beschreibt notwendige Anpassungen dieser Technik an die Serienaufnahmen der Nieren. Kapitel 8 widmet sich schließlich den angewendeten Methoden zur Berechnung der Meßkurven und automatischen Differenzierung der Nierengewebe.

Kapitel 9 zeigt den Weg zur Umsetzung der konzeptionellen Lösungen in ein Programm, dessen Bedienung durch das Installations- und Benutzerhandbuch in Kapitel 10 beschrieben wird. In Kapitel 11 werden nochmals die erarbeiteten Ergebnisse zusammengefaßt sowie ein Ausblick mit Anregungen und Erweiterungsmöglichkeiten gegeben. Damit endet diese Arbeit wieder im typisch informatischen Stil.

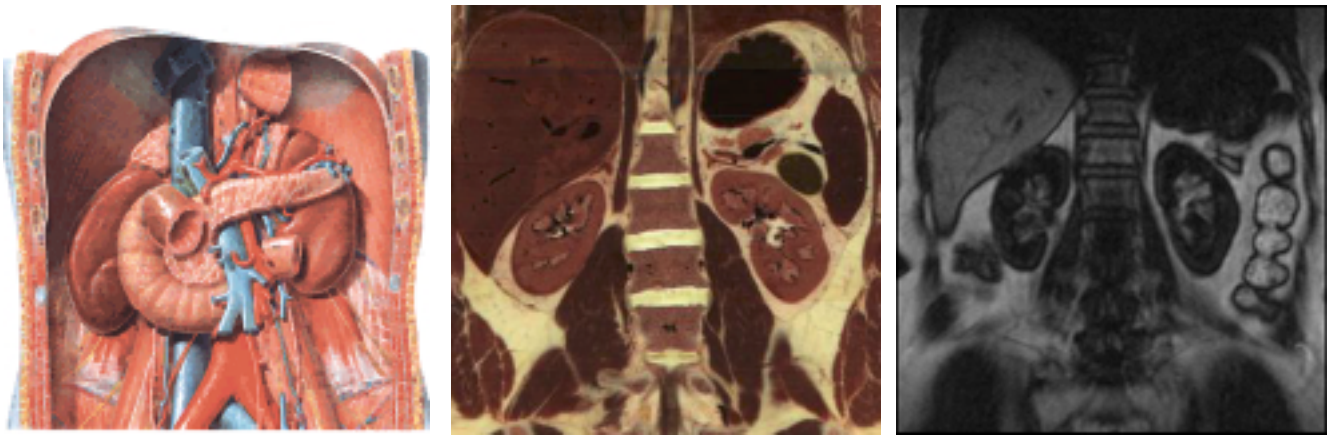
2 Anatomie und Pathophysiologie der Niere

Dieses Kapitel soll kurz das medizinische Wissen über das Organ Niere zusammenfassen, das zum Verständnis der Untersuchungsmethoden und -ergebnisse benötigt wird. Es richtet sich insbesondere an die Nicht-Mediziner unter den Lesern dieser Arbeit. Ein minimales Grundverständnis über Aufbau und Funktion des menschlichen Körpers wird aber vorausgesetzt. Medizinische Fachtermini werden weitgehend vermieden, solange sich diese durch Umschreibungen aus dem normalen Sprachgebrauch erklären lassen. Eine einfache und anschauliche Einführung in die Thematik dieses Kapitels ist in [THEW91] zu finden.

Im folgenden wird zunächst auf den Aufbau, die Lage und Funktion der Niere eingegangen. Ein Verständnis der vielfältigen biologischen und physikalischen Prozesse, die oft nur durch genaue Kenntnis des mikroskopischen Aufbaus zu erklären sind, wird im Rahmen dieser Arbeit nicht benötigt. Die wichtigsten, meist regulatorischen Leistungen der Niere sind:

- Diurese (Bildung des Harns)
- Regulation des Wasserhaushalts
- Regulation des Elektrolythaushalts
- Regulation des Säure-Base-Haushalts
- Biosynthese und Freisetzung von Hormonen (Nebennieren)

Im weiteren Verlauf werden dann krankhafte (pathologische) Veränderungen der Niere sowie deren Ursachen behandelt und somit der Bogen zu den Untersuchungsaufnahmen dieser Arbeit gespannt.



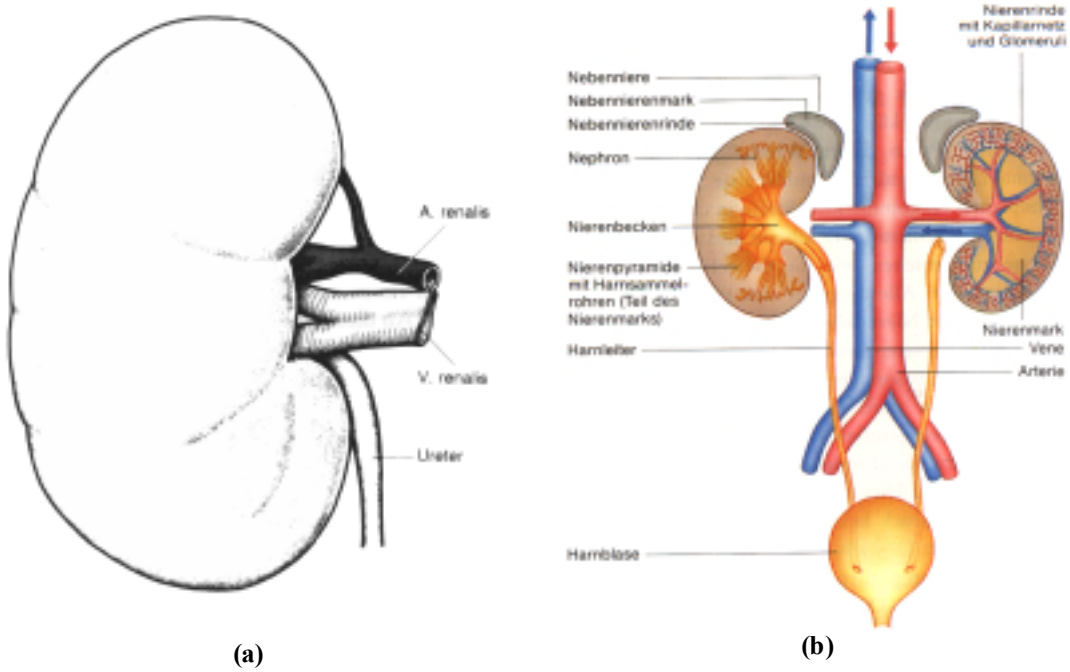
(a) (b) (c)

Abbildung 2.1: Bauchraum des Menschen. Schemazeichnung (a) [PSCH98], Anatomischer Schnitt (b) [VISI95], MRT-Aufnahme (c).

2.1 Anatomie der Niere

Lage, Form und Größe. Die paarig angeordneten Nieren liegen im Bauchraum *(Abdomen)* unterhalb des Zwerchfells beiderseits der Wirbelsäule zwischen dem Bauchfell und der hinteren Bauchwand, siehe Abbildung 2.1a. Die rechte Niere steht dabei infolge des Raumbedarfs der Leber meist etwas tiefer als die linke. Sie werden durch ein Fettpolster und eine Faszie (netzartig aufgebaute kollagene Fasern), die sie mit der benachbarten Muskulatur verbinden, in ihrer Lage gehalten. Hebt sich der Brustkorb bei der Atmung, so gleiten die Nieren - zusammen mit anderen Organen des Bauchraums – entlang der Rückenmuskulatur am *Psoas*-Muskel nach oben.

Die Gestalt der Nieren kann als bohnenförmig beschrieben werden, siehe Abbildung 2.2a (daher auch die englische Bezeichnung „kidney“ für Niere). Der konvexe Rand zeigt dabei nach außen (*lateral*), während die konkave Krümmung nach innen (*medial*) zur Wirbelsäule hin zeigt. Ungefähr in der Mitte der Konkavität befindet sich eine Vertiefung, die sogenannte Nierenpforte (*Hilus*). Durch diese treten die Nierenarterie, Nierenvene, Lymphgefäße und Nerven ein bzw. aus. Hier ist auch die Austrittsstelle des Harnleiters (*Ureter*) aus dem Nierenbecken, der den gesammelten Harn der Harnblase zuleitet.



(a) **(b)**

Abbildung 2.2: Rechte Niere in Vorderansicht (a) [THEW91], schematische Ansicht der Lage (b) [LIND89].

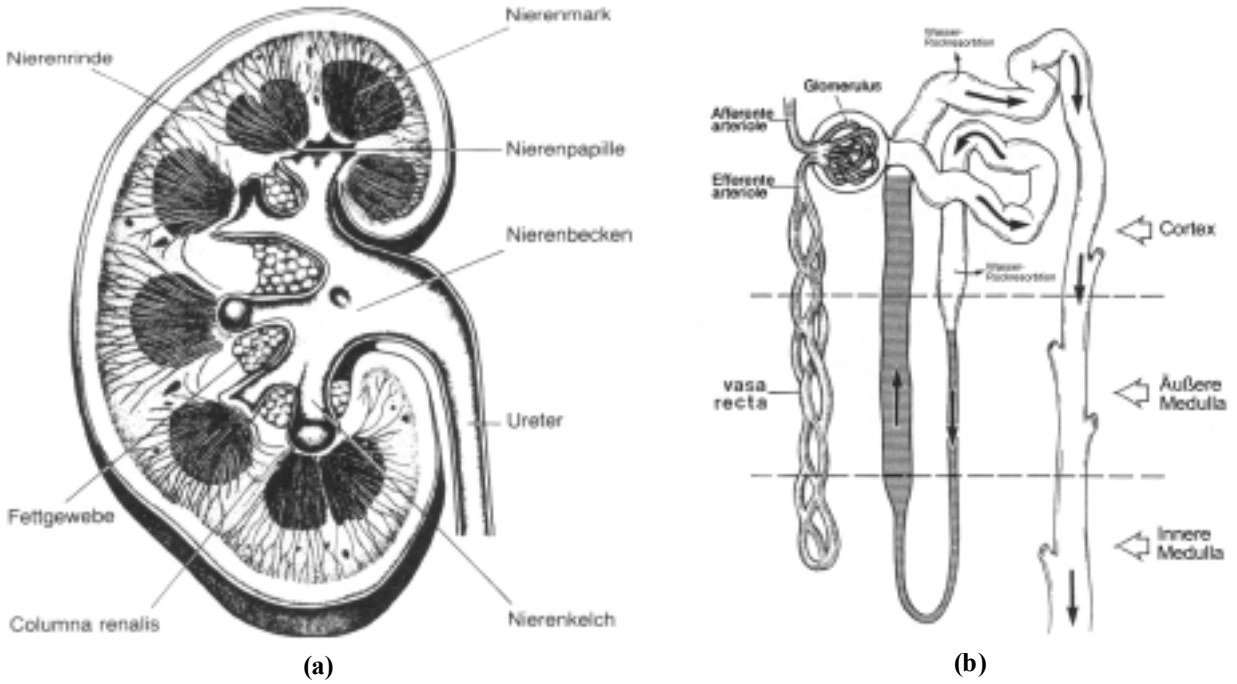
Der Längsdurchmesser einer Niere beträgt ca. 10-12 *cm*, der Querdurchmesser ca. 5-6 *cm*. Das Gewicht variiert zwischen 120 *g* und 200 *g*.

Innere Struktur. Die Niere ist von einer derben, wenig flexiblen Kapsel aus Bindegewebe überzogen. Auf einem Schnitt kann man mit bloßem Auge eine Gliederung in zwei Schichten erkennen, siehe Abbildung 2.1b und Abbildung 2.3a. Die äußere Schicht ist die Nierenrinde (*Cortex renis*) und erscheint hell und fein gekörnt. Die innere Schicht heißt Mark (*Medulla renis*), ist etwas dunkler und weist eine feine Längsstreifung auf. Das Mark ist durch mehrere Ausläufer (Säulen, *Columnae renales*) in 8-16 pyramidenförmige Lappen unterteilt, deren Spitzen zum Zentrum der Niere gerichtet sind. Diese Spitzen, die als Nierenpapillen bezeichnet werden, sind von schlauchförmigen Nierenkelchen überzogen. Sie fangen den Harn auf und leiten ihn in das Nierenbecken (*Pelvis renis*). Die morphologischen und funktionellen Einheiten, in denen die Harnbildung stattfindet, werden als *Nephrone* bezeichnet, siehe Abbildung 2.3b. Jede menschliche Niere enthält 1 bis 1,2 Millionen dieser

harnbildenden Systeme. Auf den Aufbau der Nephrone soll nicht im Detail eingegangen werden, als die wichtigsten zwei Strukturen seien jedoch die *Glomeruli* und die *Tubuli* genannt. In den *Glomeruli* wird der Primärharn abfiltriert, indem flüssige Bestandteile des Blutes durch die Gefäßwände der *Arteriolen* gepreßt werden. Der dafür benötigte Druck wird (glomerulärer) Filtrationsdruck genannt und durch den Blutdruck aufgebaut. Im *Tubulus*apparat findet die Harnkonzentration statt, indem aus dem abgepreßten Primärharn Wasser rückresorbiert wird.

Gefäßversorgung der Niere. Jede Niere wird von einem Seitenzweig der Hauptschlagader (*Aorta*), der Nierenarterie (*A. renalis*), mit Blut versorgt. Mit etwa 1,2 *l/min* oder 1700 *l/Tag* zählt die Niere zu den am stärksten durchbluteten Organen des Menschen. Bei ihrem Eintritt an der Nierenpforte verzweigt sich die Nierenarterie in die *Aa. interlobares*, die in den Säulen zwischen den Markpyramiden zur Oberfläche hin aufsteigen. An der Pyramidenbasis geben sie arkadenförmige Äste ab (*Aa. arcuatae*), aus denen dann die *Aa. interlobulares* hervorgehen. Von diesen zweigen in regelmäßigen Abständen *Arteriolen* ab, die dann die einzelnen Nierenkörperchen versorgen. Insbesondere auf die *Arteria renalis* wird im folgenden Abschnitt noch weiter eingegangen.

Das venöse Blut wird durch ein ähnlich aufgebautes System in die Nierenvene (*V. renalis*) abgeleitet, die am *Hilus* die Nieren wieder verläßt.



(a) **(b)**

Abbildung 2.3: Anschnitt der rechten Niere (a) [THEW91] und Aufbau eines *Nephrons* (b) [BENN97].

2.2 Pathophysiologie der Niere

Ebenso vielfältig und komplex wie die Ausscheidungs- und Regulationsprozesse, an denen die Nieren beteiligt sind, sind die Ursachen pathologischer Veränderungen der Nieren. Störungen des Herz-Kreislauf-Systems mit zu hohem oder zu niedrigem Blutdruck wirken sich sekundär auch auf die Nieren aus. Sinkt der mittlere arterielle Druck unter eine Grenze von 80 *mm Hg*[1], so ist die „normale" Nierendurchblutung nicht mehr gewährleistet. Harnausscheidung und andere regulatorische Mechanismen funktionieren dann nicht oder nur eingeschränkt. Bei einem Schock kann diese Situation auftreten und führt zum Bild der Schockniere, das in den allermeisten Fällen jedoch reversibel ist. Ein dauerhaft zu hoher Blutdruck hingegen kann zu einer Veränderung der Glomeruli führen

[1] Einheit zur Messung des Blutdrucks. 1 *mm Hg* = 133 *Pascal*.

(*Glomerulussklerose*). Pathophysiologisch baut sich die Niere dadurch wieder eine Druckminderung auf. Die verminderte Durchblutung führt dann aber über längere Dauer zu einem Verlust von Nierengewebe (*Parenchymschrumpfung*).

Die *Nephrone* können aber auch durch andere Ursachen geschädigt werden:

Veränderungen des *Tubulus*apparates werden ***Tubulopathien*** genannt. Meist sind dabei ein oder mehrere aktive Transportmechanismen in den Zellmembranen der *Tubuli* gestört. Sie werden noch weiter unterteilt in *proximale* und *distale Tubulopathien*, je nachdem, in welchem Bereich der *Tubuli* sich Störungen ergeben.

Unter dem Oberbegriff ***Nephritiden*** werden Entzündungen der *Nephrone* zusammengefaßt. Hier läßt sich eine Unterteilung in drei Hauptgruppen vornehmen.

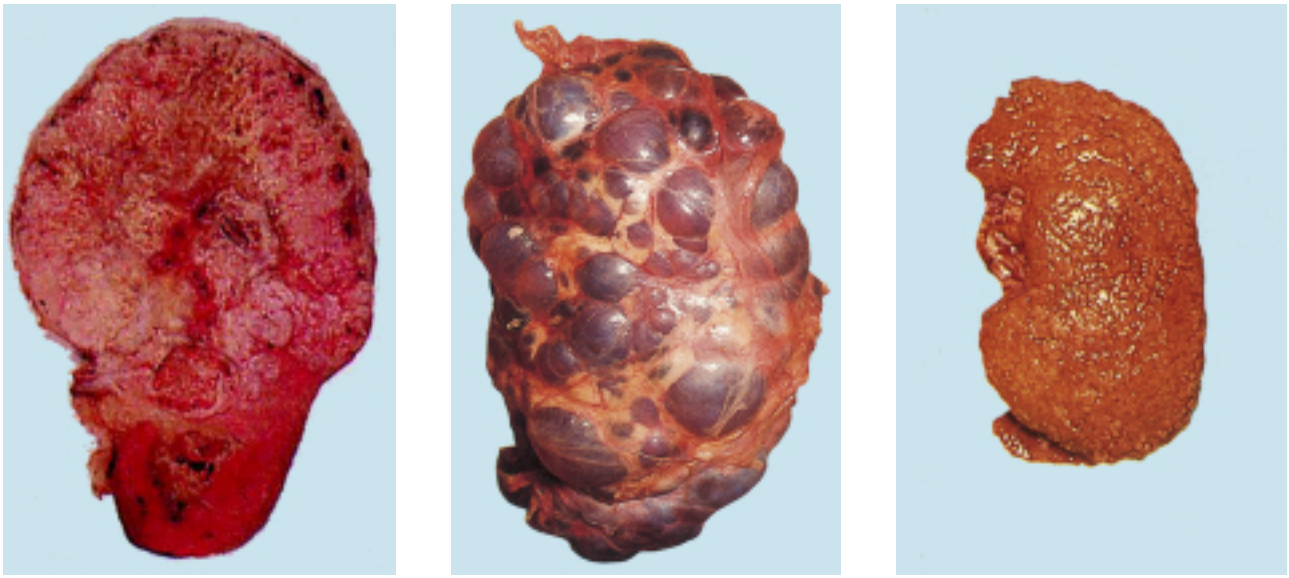
Die ***Pyelonephritiden*** kommen relativ häufig vor und sind durch eine bakteriell bedingte Entzündung des Nierenbeckens mit Beteiligung des Nierenmarks bedingt („Nierenbeckenentzündung“). Ursache sind in 80% aller Fälle *Coli*-Bakterien.

Glomerulonephritiden spielen sich anfangs vor allem im Bereich der *Glomeruli* ab. Sie werden in den meisten Fällen durch Fehl-Prozesse des Immunsystems ausgelöst.

Bei ***Interstitiellen Nephritiden*** ist primär der *interstitielle* Raum, also der Raum zwischen den Zellen, betroffen. Erst später weitet sich die *Entzündung* auf *Glomeruli*, *Tubuli* und Blutgefäße aus.

Tumoren der Niere können ebenfalls – abhängig vom Ausmaß des durch sie geschädigten Gewebes – nachhaltige Einschränkungen der Nierenfunktion bewirken. Nieren*karzinome* sind mit 85% aller auftretenden Nierentumoren die am häufigsten vorkommende Art. Männer sind – mit einem Maximum zwischen dem 45. und 75. Lebensjahr – doppelt so häufig davon betroffen wie Frauen [PSCH98]. Abbildung 2.4a zeigt ein *Karzinom* der linken Niere. Aufgrund der schon beschriebenen starken Durchblutung der Niere treten aber auch *hämatogene* (durch das Blut verbreitete) *Metastasen* von Fremd*tumoren* auf.

Nieren*zysten* treten vor allem bei älteren Patienten häufiger auf. Pathologische, durch *Zysten* bedingte Veränderungen lassen sich – ähnlich wie bei Nierentumoren – abhängig vom Ausmaß des geschädigten Gewebes feststellen. Man unterscheidet zwischen echten und falschen Nieren*zysten*, wobei letztere meist durch Gewebezerfall nach Niereninfarkten oder *Nephritiden* bedingt sind.



(a) **(b)** **(c)**

Abbildung 2.4: Nierenkarzinom (a) [PSCH98] , Zystenniere (b) [ROCH96], Schrumpfniere (c) [PSCH98].

Das genetisch vererbte und mit einer Wahrscheinlichkeit von 1:1000 relativ häufig auftretende Krankheitsbild der Zystenniere sei hier auch noch erwähnt. Dabei wird die Niere durch multiple Zystenbildung meist sehr stark in ihrer Funktion eingeschränkt, siehe Abbildung 2.4b.

Die meisten Nierenerkrankungen können, unabhängig von ihrer Ursache, in einer chronischen Niereninsuffizienz enden. Die chronische Niereninsuffizienz ist eine fortschreitende Funktionseinschränkung, die auf den Verlust funktionstüchtiger *Nephrone* zurückzuführen ist. Durch den progressiven Parenchymverlust entwickelt sich schließlich das Bild der Schrumpfniere, siehe Abbildung 2.4c. Die Funktionseinschränkung der Nieren wirkt sich natürlich nachhaltig auf alle Regelkreisläufe des menschlichen Organismus aus, an denen die Niere beteiligt ist. Insbesondere die Störung des Wasserhaushaltes und damit der Blutvolumenregulierung führt in vielen Fällen zu einem erhöhten Blutdruck [THEW91].

2.3 *Renovaskuläre Hypertonie*

2.3.1 Blutdruck und Blutdruckregulierung

Die Mechanismen des menschlichen Körpers zur Regulierung des Blutdrucks sind äußerst komplex. Verschiedene, ineinandergreifende Abläufe steuern die kurzfristige, mittelfristige oder langfristige Blutdruckregulation. Im Rahmen dieser Arbeit und im Zusammenhang mit dem Bluthochdruck (Hypertonie) interessieren insbesondere die Mechanismen zur langfristigen Regulation. Abbildung 2.6 gibt einen Überblick über die Zusammenhänge.

Sie beruhen vor allem aus einer Anpassung des extrazellulären Flüssigkeitsvolumens – und damit gekoppelt des Blutvolumens – an veränderte Kreislaufsituationen. Das Stellglied dieses Regelkreises ist die Flüssigkeitsausscheidung der Nieren.

Renale Volumenregulation.

Arterielle Blutdrucksteigerungen lösen eine vermehrte Harnausscheidung aus. Die Folge ist eine Reduktion des extrazellularen Flüssigkeitsvolumens und des Blutvolumens, wodurch der Rückfluß zum Herzen vermindert wird, siehe Abbildung 2.6. Damit kommt es (aufgrund einer Abnahme des Herzzeitvolumens) zu einer Senkung des zunächst erhöhten, arteriellen Blutdrucks. Auf diesem Wege kann z.B. bei bestimmten Formen der Hypertonie der Blutdruck durch die Gabe von harntreibenden Substanzen (Diuretika) gesenkt werden. Eine primäre Erniedrigung des Blutdrucks löst dagegen eine verminderte Harnausscheidung (H_2O-*Retention*) aus. Vermittelt wird der gesamte Regelprozeß durch die Hormone *Aldosteron* und *Adiuretin*.

Renin-Angiotensin-Aldosteron-Mechanismus

Bei einer Minderdurchblutung wird in der Niere der Botenstoff *Renin* freigesetzt, der zu einer Bildung von *Angiotensin II* führt und damit die Harnausscheidungsrate der Niere beeinflußt. *Angiotensin II* stimuliert die *Aldosteron*-Ausscheidung durch die Nebennierenrinden. Unter dem Einfluß von *Aldosteron* wird die Rückgewinnung (Rückresorption) von *Natrium* aus dem *Tubulus*system der Niere gefördert und damit letztendlich weniger Flüssigkeit ausgeschieden. Das *Renin-Angiotensin-Aldosteron*-System bewirkt also, daß das Blutvolumen bei Blutdruckabfall erhöht und bei Blutdruckanstieg reduziert wird.

Ein weiterer wichtiger Zusammenhang besteht zwischen dem *Renin-Angiotnesin*-Mechanismus und Nierenarterien*stenosen*, siehe Kapitel 2.3.3. Liegt eine *Stenose*, also eine Verengung des zuleitenden Blutgefäßes zur Niere vor, so sinkt der Blutdruck im weiteren Verlauf des Gefäßes. Damit sinkt auch der Filtrationsdruck in den *Glomeruli* und es kann nicht mehr die erforderliche Menge Primärharn abgepreßt werden. Dem versucht die Niere durch eine gesteigerte *Renin*-Ausschüttung entgegenzuwirken. Über den *Renin-Angiotensin*-Regelkreis wird in der Gefäßmuskulatur der kleinen Arterien eine Kontraktion ausgelöst, siehe auch Abbildung 2.5c. Diese *Vaso-konstriktion* soll sich auf

die *efferenten* (vom *Glomerulum* wegführenden) Gefäße auswirken und dadurch im *Glomerulum* den erforderlichen Filtrationsdruck wiederherstellen. Durch diesen Regelkreis gelingt es der Niere, auch ausgeprägten *Stenosen* der Nierenarterien entgegenzuwirken[2].

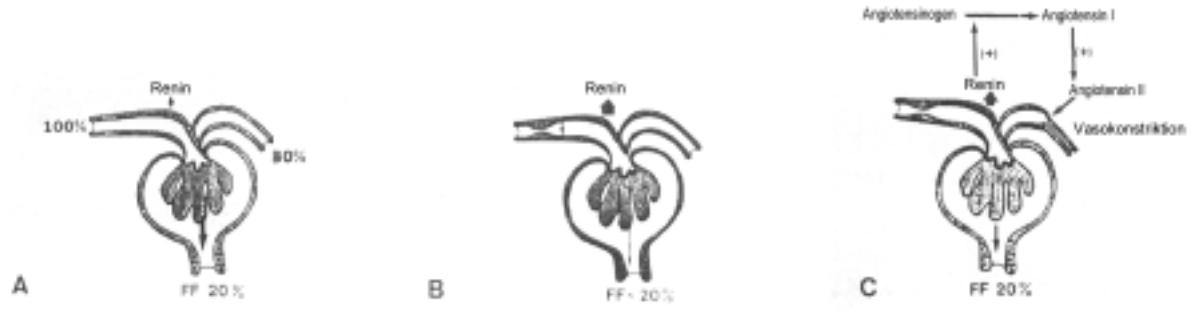


Abbildung 2.5: Wirkungsweise von *Renin* an den *Glomeruli* [BUBE93].

Da die *vasokonstriktorischen* Einflüsse von *Angiotensin* nicht organspezifisch beschränkt sind, betreffen sie bei anhaltender Aktivierung alle Arterien des Blutkreislauf und wirken sich damit sekundär blutdrucksteigernd aus. Durch den zuvor beschriebenen Zusammenhang mit der Volumenregulierung wird die blutdrucksteigernde Wirkung weiter unterstützt.

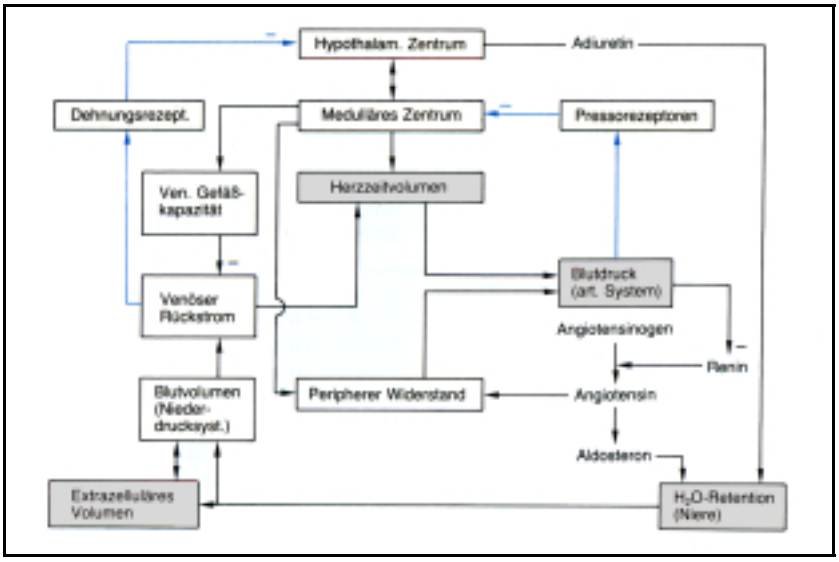


Abbildung 2.6: Die komplexen Zusammenhänge der menschlichen Blutdruckregulierung [THEW91].

2.3.2 Hypertonie

Als Bluthochdruck (*Hypertonie*) bezeichnet man jede die Norm überschreitende, anhaltende Steigerung des arteriellen Blutdrucks. Nach der WHO[3]-Definition besteht eine *Hypertonie*, wenn in körperlicher Ruhe – unabhängig vom Alter – der arterielle (*systolische*) Blutdruck mehr als 160 *mm Hg* und der venöse (*diastolische*) Blutdruck mehr als 95 *mm Hg* beträgt. Der beobachtungsbedürftige Grenzbereich liegt *systolisch* zwischen 140 und 160 *mm Hg* und *diastolisch* zwischen 90 und 95 *mm Hg*, siehe auch Abbildung 2.7. Man unterscheidet weiter zwischen der primären und sekundären *Hypertonie*. Die am häufigsten anzutreffende Form ist die primäre *Hypertonie* (ca. 90% aller Fälle). Es gilt heute als weitgehend gesichert, daß vererbungsbedingte (*hereditäre*) Komponenten eine Rolle dabei spielen [THEW91]. Bei der in ca. 10% aller Fälle vorliegenden sekundären Form stellt die

[2] Erst ab einem *Stenose*grad von mehr als 90% funktioniert die Regulierung nicht mehr zuverlässig, siehe [BUBE93].
[3] Welt-Gesundheits-Organisation (engl. WHO für „World Health Organisation“)

Blutdruckerhöhung ein Folgesymptom einer anderen Primärerkrankung dar. Zirka die Hälfte der sekundären *Hypertonien* sind auf die im folgenden beschriebenen Nierenarterien*stenosen* zurückzuführen und werden als *renovaskuläre Hypertonie* bezeichnet. Sie wird durch einseitige oder beidseitige Einengung der Nierenarterien verursacht. Die *stenose*bedingte Mangeldurchblutung der Niere führt zu einer gesteigerten *Renin*freisetzung, so daß der *renovaskuläre* Bluthochdruck auf dem zuvor geschilderten Weg über das *Renin-Angiotensin-Aldosteron*-System zustande kommt.

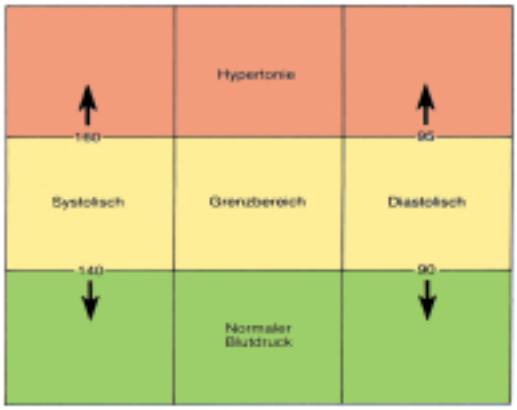


Abbildung 2.7: Hypertonischer Bereich und Grenzbereich des Blutdrucks [THEW91].

Bei zirka 50 Millionen *Hypertonikern* in den USA sind das ca. 2 Millionen Menschen [KING95]. Bei gleicher Verteilung innerhalb der deutschen Bevölkerung gibt es schätzungsweise 800.000 Fälle.

2.3.3 Nierenarterien*stenose*

Die Nierenarterien*stenose* (NAS) verursacht ebenso wie die in Kapitel 2.2 beschriebenen Krankheitsbilder pathologische Veränderungen der Nieren, jedoch wirkt sie nicht direkt auf das Nierengewebe ein. Mit dem Begriff *Stenose* wird allgemein die Einengung eines Gefäßes beschrieben. *Stenosen* entstehen zunächst durch Ablagerung von Bestandteilen des Blutes (*fibröse Plaques*) an der Innenseite des Gefäßes, die sich im weiteren Verlauf langsam vergrößern und schließlich durch Einlagerung von Kalksalzen verhärten. Dabei kommt es teilweise sogar zum Verschluß der Arterien, z.B. beim Herzinfarkt oder beim Schlaganfall. Prinzipiell stellt eine *Stenose* der zuführenden Blutwege ein Problem für alle Organe dar, die auf eine regelmäßige Durchblutung angewiesen sind.

Nierenarterienstenosen lokalisieren sich meistens kurz nach dem Abgang der *A. renalis* aus der Hauptschlagader. Häufig ist im weiteren Verlauf eine Erweiterung (*Dilatation*) des Gefäßes direkt hinter der verengten Stelle zu finden, verursacht durch Wirbel im Blutstrom aufgrund der Ablagerung. Infolge der chronischen Minderdurchblutung kommt es, wie beschrieben, zu Fehlregulierungen des Blutdrucks; später dann zur Niereninsuffizienz und schließlich zur Schrumpfniere.

Die NAS ist, auch durch die Lebensgewohnheiten in der modernen Wohlstandsgesellschaft bedingt, mit zunehmender Häufigkeit Ursache für degenerative Veränderungen der Nieren. Die beschriebene, durch Ablagerungen bedingte *arteriosklerotische* Nierenarterien*stenose* tritt gehäuft im 5.-6. Lebensjahrzehnt auf. Weitaus seltener und vor allem bei Frauen im dritten Lebensjahrzehnt kommt die *fibromuskuläre* Nierenarterien*stenose* vor.

Diagnostik von Nierenarterienstenosen

Zur Diagnose einer NAS wird als Standardverfahren die Nieren*angiographie*t, ein invasives Verfahren mit Strahlenexposition für den Patienten, eingesetzt. Wie schon in der Einleitung erwähnt, findet hier die Magnetresonanztomographie vermehrt ihren Einsatz. Durch die Gabe eines Kontrastmittels, kombiniert mit speziellen Aufnahmeparametern und spezieller Aufnahmegeometrie, können Ergebnisse erzielt werden, die mit der *Angiographie* vergleichbar sind. Abbildung 2.8 zeigt in (a) das

Ergebnis einer Nierenangiographie, durchgeführt als digitale Subtraktionsangiographie (DSA). Bild (b) und (c) zeigen 3D-Angio-MRT-Aufnahmen ohne und mit Befund einer Stenose. In der Vergrößerung in (d) ist eine (eingangs erwähnte) *poststenotische Dilatation* der Nierenarterie zu erkennen. WASSER et al. verglichen in einer Arbeit von 1997 die Anwendbarkeit der MRT mit der (herkömmlichen) digitalen Subtraktionsangiographie als Diagnosemethode von Nierenarterienstenosen [WASS97].

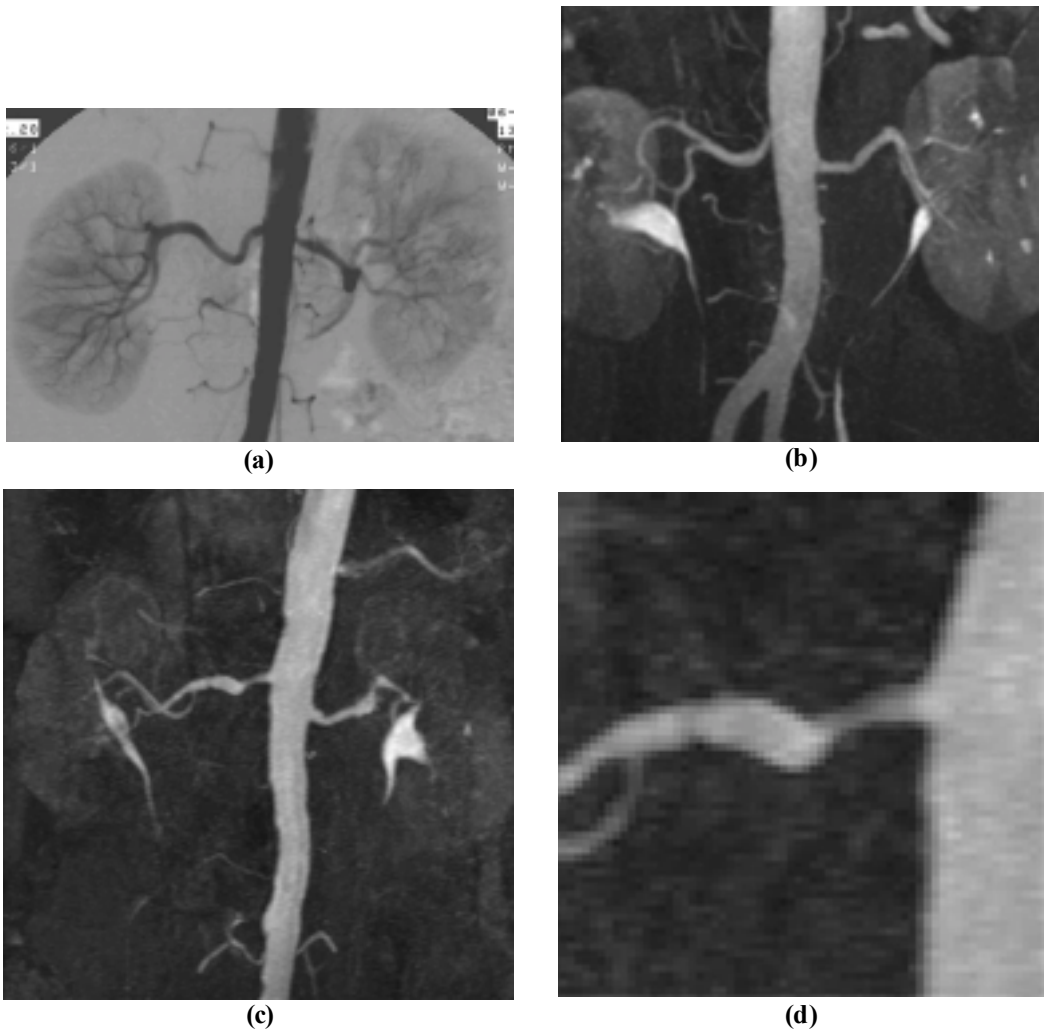


Abbildung 2.8: Digitale Röntgen-Subtraktionsangiographie (a), MRT-3D-Angiographie-Aufnahme ohne Befund (b), MRT-3D-Angiographie-Aufnahme mit Stenose (c) und Vergrößerung der poststenotischen Dilatation (d).

Diagnostik der Nierenfunktion

Hinsichtlich eines diagnostisch abzuklärenden Zusammenhangs zwischen vorliegender NAS und erhöhtem Blutdruck kann eine Untersuchung der Nierenfunktion hilfreich sein. Hierbei können verschiedene Verfahren eingesetzt werden, um über einen bestimmten Zeitraum quantitative Aussagen über die Nierenperfusion zu machen.

Die Messung der *glomerulären* Filtrationsrate ist eines der Standardverfahren zur Überprüfung der Nierenperfusion. Wie bereits erwähnt, wird durch den Blutdruck in den *Glomeruli* der Niere ein Filtrat abgepreßt, der sog. Primärharn. Das Filtratvolumen, das pro Zeiteinheit in sämtlichen *Glomeruli* entsteht, wird als *glomeruläre* Filtrationsrate (GFR) bezeichnet. Bei einem 20jährigen Mann beträgt sie etwa 125 *ml/min*, bei einer gleichaltrigen Frau ca. 110 *ml/min*. Zum Alter hin nimmt die GFR auch

beim Nierengesunden kontinuierlich ab; bei Menschen im Alter von 70 Jahren beträgt sie nur noch etwas die Hälfte der oben genannten Werte.

Die Nierenfunktion wird anhand der GFR gemessen und kann mit Hilfe der *Creatinin*- oder (genauer, aber schwieriger durchzuführen) mit der *Inulin*-Clearance festgestellt werden. Bei letztgenannter Methode wird dem Patienten der Fruchtzucker (*Fructosepolysaccharid*) *Inulin* verabreicht und anschließend die Konzentration des Stoffes im Harn bestimmt. Dieser wird über einen festgelegten Zeitraum gesammelt. Die Ausscheidungsrate ergibt sich aus dem Produkt der Inulinkonzentration im Urin und dem Harnzeitvolumen.

Um gesonderte Werte für linke und rechte Niere zu bekommen, müssen andere Verfahren eingesetzt werden. Die Nierenfunktions-Szintigraphie, die Aktivitätskurven beider Nieren nach Gabe radioaktiv markierter Substanzen aufnimmt, wird hier bereits seit Anfang der achtziger Jahre als Standard-verfahren eingesetzt. Im Laufe der Zeit wurden einige (auf der Szintigraphie basierende) erweiterte Verfahren vorgestellt, die zusätzlich Medikamente (ACE-Hemmer) verwenden, die direkt in das *Renin-Angiotensin*-System eingreifen [BUBE93]. Die nuklearmedizinische Untersuchungsmethode bleibt jedoch immer mit einer Strahlenexposition für den Patienten verbunden.

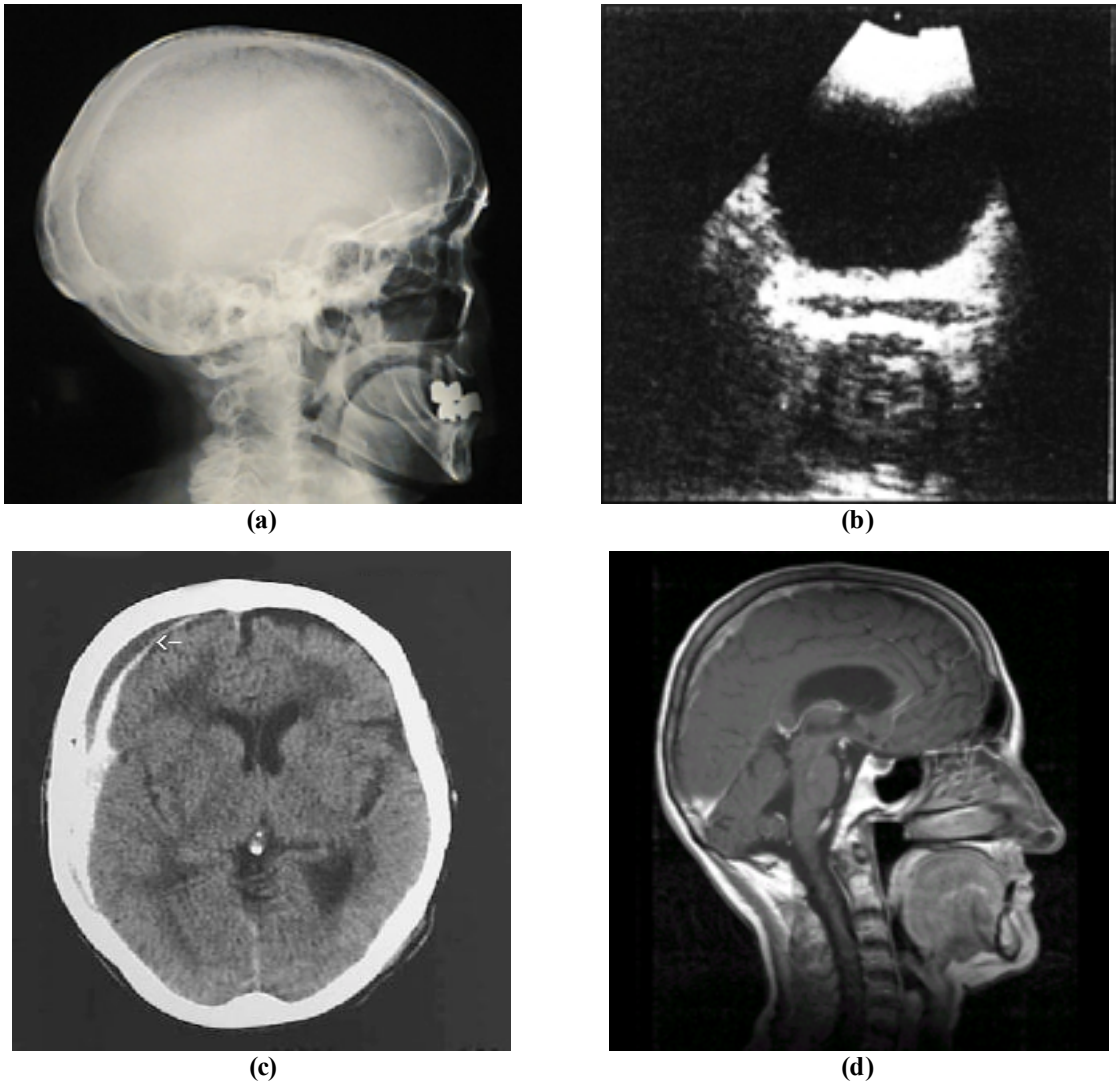
Die Magnetresonanztomographie als nicht-invasive Untersuchungsmethode ohne Strahlenexposition hat seit kurzer Zeit auch hier Einzug gehalten. Die Kombination schneller Aufnahmesequenzen (z.B. Turbo-Flash, vergleiche Kapitel 3.2.1) und Auswertung einer dynamischen Kontrastmittelpassage durch die Nieren ermöglicht eine Diagnosestellung. MILLER et al. haben 1999 in einer Studie gezeigt, daß die Diagnostik mit im praktischen Einsatz weit verbreitetem MRT-Geräten einer Feldstärke von 1,0 *Tesla*[4] mittels einer Turbo-Flash-2D-Sequenz sehr gut durchführbar ist [MILL99]. Dabei zeigten das aorto-renale Anflutungsverhalten sowie der maximale Signalanstieg der Nierenrinde bei NAS eine signifikante Seitendifferenz. STATTAUS et al. stellten etwa zeitgleich ein ähnliches Verfahren vor, mit dem ein verzögerter Anstieg der Kontrastmittelaufnahme bei vorliegender NAS nachzuweisen war [STAT99]. Die in dieser Arbeit behandelten Meßreihen wurden mit dieser Methode erstellt.

[4] Einheit zur Messung der magnetischen Feldstärke (= Flußdichte). 1 *Tesla* = 1 $\frac{Vs}{m^2}$.

3 Schnittbilddiagnostik

In der modernen Medizin sind bildgebende Verfahren zur Unterstützung der Diagnose nicht mehr wegzudenken. Sie sind zumeist das Produkt interdisziplinärer Forschungsansätze von Medizin und Technik. Während der Informatiker und / oder Ingenieur sich bei der Einteilung und Einordnung an den dahinter liegenden physikalischen Prinzipien orientiert, stehen für den Mediziner andere Gesichtspunkte im Vordergrund, wie z.B. Einsatzmöglichkeit und Diagnosegenauigkeit [LEHM97].

Aus technischer Sicht lassen sich vier bildgebende Verfahren unterscheiden, die heute in der Routinemedizin verstärkt eingesetzt werden. Das sind das klassische Röntgen, die Sonographie (US = Ultraschall), die Computertomographie (CT) und die Magnetresonanztomographie (MRT), auch Kernspintomographie genannt. Darüber hinaus gibt es noch viele weitere Formen der medizinischen Bildgebung. Hier soll jedoch nur auf das zur Schnittbilddiagnostik eingesetzte Verfahren der MRT eingegangen werden, da das dieser Untersuchung zugrundeliegende Bildmaterial aus einem Kernspintomographen stammt. Ein kurzer Einblick in die physikalischen Grundlagen soll ebenfalls vermittelt werden.



(a) **(b)**

(c) **(d)**

Abbildung 3.1: Röntgenaufnahme (a), Ultraschallaufnahme einer Harnblase (b) [LEHM97], CT-Aufnahme eines Kopfes (c), MRT-Aufnahme eines Kopfes (d).

Die Verfahren haben sehr unterschiedliche Eigenschaften in Bezug auf die Darstellung verschiedener Gewebearten und -funktionen. Das klassische Röntgen ist ein preiswertes Verfahren und eignet sich insbesondere zur Darstellung von Knochen. Es hat allerdings den Nachteil, daß gemessene Volumina auf eine zweidimensionalen Ansicht reduziert werden. Genau in diesem Punkt unterscheiden sich die Tomographieverfahren CT und MRT wesentlich davon. Unter einer Tomographie versteht man im allgemeinen die Abbildung einer zweidimensionalen Schicht eines dreidimensionalen Objektes, also ein Schnittbildverfahren. Die Computertomographie eignet sich, da sie auch auf Röntgenstrahlung basiert, ebenfalls zur Darstellung von Knochen. Da hier aber eine abzubildende Schicht aus einem Volumen ausgewählt werden kann, können mit der CT auch Flächen und Volumen sehr genau gemessen werden. Letzteres gilt auch für die Magnetresonanztomographie. Aufgrund einer völlig anderen Funktionsweise weisen MRT-Bilder einen sehr hohen Weichteilkontrast auf.

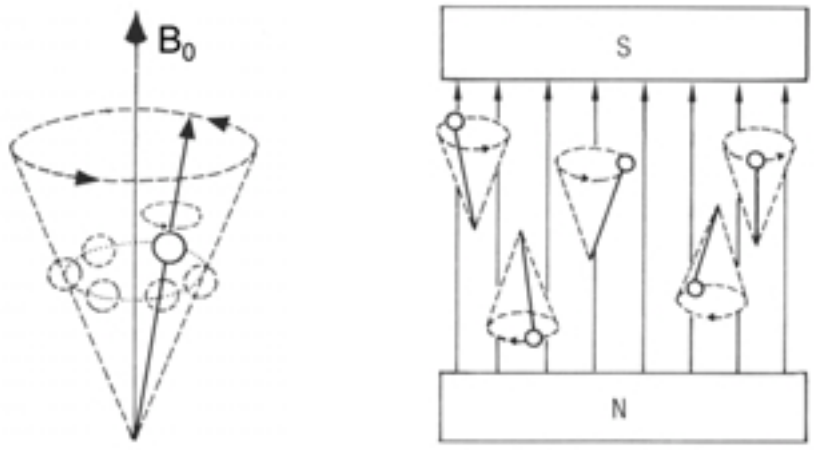
Das Ultraschallverfahren eignet sich aufgrund seiner Echtzeitfähigkeit insbesondere zur Darstellung von Organfunktionen. Es ist ein sehr preiswertes und überall verfügbares Verfahren, das allerdings keine hohe Detaildarstellung erlaubt und ebenso wenig exakt reproduzierbar ist. Tabelle 3.1 faßt die Darstellungsmöglichkeiten der vier wichtigsten bildgebenden Verfahren zusammen.

	Röntgen	CT	MRT	US
Knochen	++	++	+	–
Weichteile	–	–	++	+
Gefäße	++	++	++	++
Funktionen	–	–	++	++
Volumina	–	++	++	+

Tabelle 3.1: Darstellungsmöglichkeiten der 4 wichtigsten bildgebenden Verfahren in der Medizin nach [LEHM97].

3.1 Magnetresonanztomographie

Bei der Magnetresonanztomographie oder auch Kernspintomographie macht man sich den Eigendrehimpuls – oder Spin – der Atomkerne mit ungerader Anzahl an Nukleonen zunutze. Die Kerne dieser Atome gleichen, schematisch betrachtet, rotierenden Kreiseln (Abbildung 3.2a), deren Drehachsen beliebig im Raum orientiert sind. Bringt man sie aber in ein starkes äußeres Magnetfeld, so richten sich die Rotationsachsen parallel oder antiparallel zu diesem Feld aus (Abbildung 3.2b).

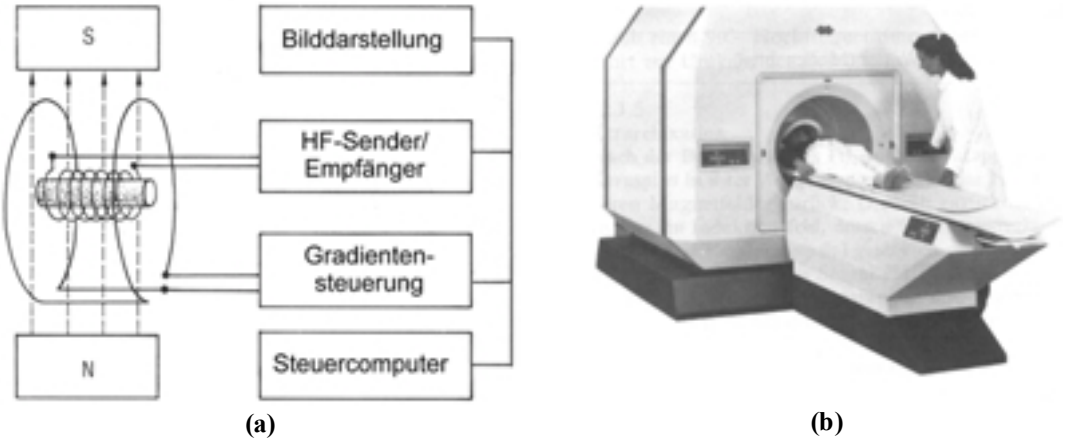


(a) **(b)**

Abbildung 3.2: Kernspin und Präzession [LAUB90] (a), Ausrichtung der Kernspins im Magnetfeld [LAUB90] (b).

Bei 37°*C* und einer magnetischen Flußdichte von 1,0 *Tesla* ist die Anzahl der parallel ausgerichteten Atome aus energetischen Gründen entsprechend der BOLTZMANN-Verteilung etwa ein 7 millionstel höher als die der antiparallel ausgerichteten. Da sich jeweils gleiche Anzahlen gegensätzlich ausgerichteter Atome in ihrer Wirkung aufheben, entstehen die Kernspinbilder gerade nur durch diese geringe Zahlendifferenz (die aber realtiv zur Anzahl der gesamten Elektronen ausreicht).

Nach der Ausrichtung durch das äußere Magnetfeld führen die Kerne zusätzlich zu ihrer Eigenrotation eine Kreiselbewegung (Präzession) um die Richtung des Magnetfeldes aus (Abbildung 3.2a). Die Frequenz dieser Präzessionsbewegung wird auch als Larmor- oder Resonanzfrequenz bezeichnet. Grundlegend für die Kernspintomographie ist dabei, daß die Frequenz mit Zunahme der Feldstärke auch linear ansteigt, damit gibt es für jede Feldstärke genau eine Larmorfrequenz.



(a) **(b)**

Abbildung 3.3: Blockschaltbild eines MRT [LAUB90] (a), MRT-Gerät (b) [LAUB90].

Setzt man die Atome nun zusätzlich einer hochfrequenten elektromagnetischen Strahlung (HF) aus, deren Frequenz genau der Larmorfrequenz entspricht, nehmen sie aus diesem Feld Energie auf. Dadurch wird der Winkel zwischen Magnetfeld und Rotationsachse zunehmend größer, etwa so, als ob man einen trudelnden Kreisel immer wieder anstößt. Nach Abschaltung des HF-Pulses haben die Atome die Tendenz, wieder ihre ursprüngliche Lage einzunehmen und senden dabei ihrerseits eine elektromagnetische Welle der Larmorfrequenz aus. Diese kann über ein Antennensystem empfangen werden und wird schließlich als Kernspin-Resonanz-Signal registriert.

Um nun ein ortsaufgelöstes Bild zu erzeugen, sorgt man durch drei senkrecht zueinander orientierte Elektromagnete (sogenannte Gradienten-Magnete) dafür, daß an jedem Raumpunkt innerhalb des Kernspintomographen ein anderes Magnetfeld herrscht. Damit ist auch die Larmorfrequenz an jedem Punkt unterschiedlich, so daß man durch geeignete Wahl der HF-Frequenz gezielt die Kerne in einem bestimmten Raumpunkt anregen kann.

Durch eine geeignete zeitliche Abfolge der Schaltung der Gradienten (Schichtselektionsgradient, Phasenkodiergradient und Auslesegradient), der HF-Impulse und der Empfängerschaltungen kann nun ein beliebig orientiertes Schnittbild des Körpers angefertigt werden. Eine solche Schaltfolge wird auch Pulssequenz genannt.

Durch verschiedene Aufnahmeparameter (u.a. Frequenz- und Gradienteneinstellungen) am Kernspintomographen lassen sich eine Vielzahl von Aufnahmetypen erzeugen. Diese unterscheiden sich hinsichtlich der Darstellungsart verschiedener Gewebearten. Daher sind häufig die Aufnahmeparameter durch die Art der zu klärenden medizinischen Fragen gesteuert. Tabelle 3.2 gibt einen kleinen Überblick über die Möglichkeiten der Darstellung verschiedener Gewebe mit Hilfe der Magnetresonanztomographie.

Signalverhalten	T_1-gewichtetes Bild	T_2-gewichtetes Bild
signalreich	Fett	Liquor, Ödem, Flüssigkeit
mittlere Intensität	Lymphknoten, Muskulatur, Knochenmark, Knorpel	Fett, Knochenmark
signalarm	Liquor, Zysten, Flüssigkeit, Verkalkungen, Kortikalis[5]	Kortikalis, Verkalkungen, Muskulatur

Tabelle 3.2: Signalintensität verschiedener Strukturen in T_1- und T_2-gewichteten Bildern [UHLE96].

Bei der in dieser Arbeit behandelten funktionellen Perfusions-untersuchung der Nieren ist der Verlauf einer Kontrastmittelpassage durch die Nieren von Interesse. Je höher die zeitliche Auflösung ist, d.h. je schneller die Aufnahmen hintereinander erzeugt werden können, desto präziser wird die Diagnose. Gleichzeitig muß aber auch eine Mindestanforderung an die Auflösung der Bilder existieren, damit die verschiedenen, räumlich eng beieinanderliegenden Gewebestrukturen der Nieren differenzierbar bleiben. Die sogenannten gespoilten Gradienten-Echo-Sequenzen haben sich hierfür als geeignet herausgestellt [KNOO97], [MILL99]. Im folgenden soll nun kurz auf die verschiedenen Aufnahmetechniken der Magnetresonanztomographie eingegangen werden. Eine anschauliche Einführung ist in [PYKE81] zu finden, für eine ausführlichere Darstellung der Materie wird [LISS90] oder [SIEM92] empfohlen.

3.2 MRT-Aufnahmetechniken

Allgemein lassen sich die verschiedenen Aufnahmetechniken der Magnetresonanztomographie in zwei große Gruppen unterteilen, die konventionellen Spin-Echo-Techniken und die Gradienten-Echo-Techniken. Um den Unterschied zu verstehen, soll noch einmal kurz auf die oben genannten HF-Impulse zur Anregung der Kernspins eingegangen werden. Das anschauliche Modell des rotierenden Kreisels (Abbildung 3.2a) wird dabei auf sein Verhalten bei der Einstrahlung von 90° - und 180° HF-Impulsen hin untersucht. Man stelle sich den Kreisel in einem dreidimensionalen Koordinatensystem vor, wobei die Kreiselspitze im Nullpunkt liegt und die Kreiselhauptachse dem Magnetisierungsvektor M des Protons entspricht. Dieser zeigt im statistischen Mittel (über die Anzahl der Kernspins) in die Richtung des homogenen Magnetfeldes des Tomographen und liegt auf der z-Achse des Koordinatensystems. Je nach Stärke und Dauer des HF-Impulses wird M nun aus seiner ursprünglichen Lage gekippt, die Winkelgröße bezeichnet dabei die Ablenkung der Kreiselhauptachse von der z-Achse des Koordinatensystems. Abbildung 3.4 soll diesen Sachverhalt für einen 90°- und einen 180°-Impuls verdeutlichen.

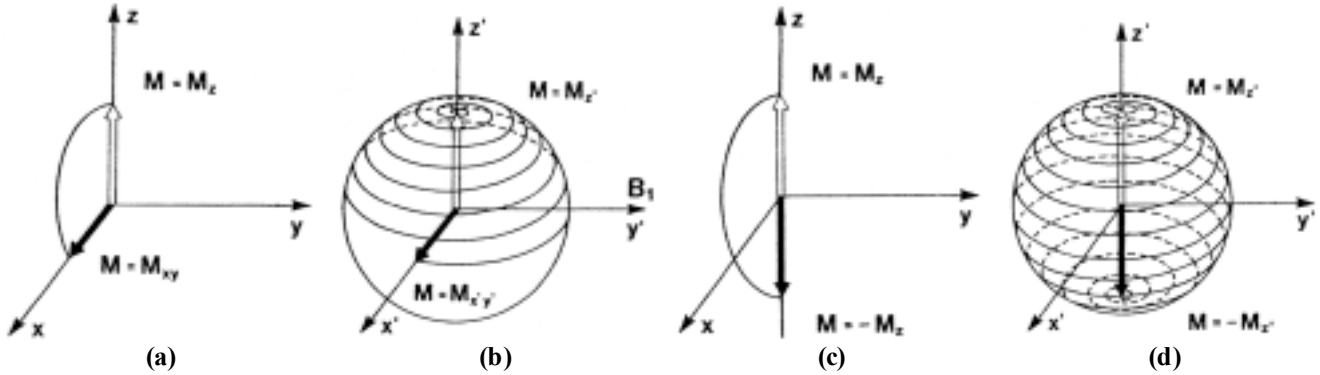


Abbildung 3.4: Auswirkung eines 90°- und eines 180°-Impulses auf den Kernspin [LISS90]. Schematische Darstellung eines 90°-impuls (a) und im rotierenden Koordinatensystem (b). Bilder (c) und (d) entsprechen einem 180°-Impuls.

[5] Knochengewebe (Knochenwand)

Wie eingangs erwähnt, benötigen die Atomkerne nach Abschaltung des Impulses nun einige Zeit, um in ihre ursprüngliche Position zurückzukehren. Diesen Vorgang nennt man Relaxation, wobei die Zeit, die die Protonen dafür benötigen, als Relaxationszeit bezeichnet wird. Die dabei entstehenden elektromagnetischen Wellen werden gemessen und sind letztendlich für die Erzeugung der Bilder verantwortlich. Je nach Art des dargestellten Gewebes, des Gradientenbetrages und der HF-Frequenz ist die Dauer der Relaxation unterschiedlich lang. Man unterscheidet nun weiter zwischen zwei verschiedenen Relaxationsprozessen.

Nach dem Anregungsimpuls ist die *z*-Komponente des Magnetisierungsvektors (M_z) minimal, um dann exponentiell auf den ursprünglichen Wert anzuwachsen. Der Anstieg der Exponentialfunktion ist dabei durch eine „Erholungszeit" T_1 gekennzeichnet, die man als Thermische Relaxationszeit oder Spin-Gitter-Relaxationszeit bezeichnet. Sie hängt davon ab, wie lange das angeregte Spin-System braucht, um mit seiner Umgebung (dem „Gitter") wieder in das thermische Gleichgewicht zu kommen.

Die Kernspins treten aber nicht nur mit ihrer Umgebung, sondern auch untereinander in Wechselwirkung, so daß neben der Spin-Gitter-Relaxationszeit zusätzlich eine Spin-Spin-Relaxationszeit T_2 auftritt. Sie bewirkt, daß das Induktionssignal bereits abklingt, bevor das Spin-System sein thermisches Gleichgewicht wiedererlangt hat. Durch die Wechselwirkungen zwischen den magnetischen Momenten benachbarter Kerne richten sich die einzelnen Momente so aus, daß sie sich „vorzeitig" kompensieren. Dieser – ebenfalls exponentielle – Abfall ist durch die Spin-Spin-Relaxationszeit T_2 charakterisiert.

In einem exakt homogenen Magnetfeld klingt ein Induktionssignal mit dem Zeitfaktor T_2 ab. In der Praxis zerfällt es jedoch schneller, denn reale Felder weisen immer Inhomogenitäten auf, die die exponentielle „Dämpfung" der Signalamplitude beschleunigen. Der zugehörige Zeitfaktor wird im allgemeinen mit T_2* bezeichnet.

Die Bildergebnisse der verschiedenen Aufnahmetechniken sind nun unterschiedlich abhängig von der T_1- und T_2-Zeit. Die konventionellen Spin-Echo-Sequenzen beruhen auf den Messungen beider Zeiten und müssen den Nachteil längerer Akquisitionszeiten in Kauf nehmen, da aufgrund des langsameren Abfalls der T_1-Zeit die Repetitionszeit im Sekundenbereich liegt. Dagegen basiert die Bildgewinnung bei den Gradienten-Echo-Sequenzen hauptsächlich auf der T_2*-Zeit und ist daher schneller.

3.2.1 Turbo-Flash-Sequenzen

Die Flash-Aufnahmen sind ein klassischer Vertreter der Gradienten-Echo-Technik. Turbo-Flash-Sequenzen besitzen – wie der Name schon sagt – einen noch schnelleren Bildgewinnungzyklus. Das verwendete Akronym für gespoilte Gradienten-Echo-Sequenzen ist dabei von Hersteller zu Hersteller unterschiedlich. Als Überschrift für dieses Kapitel wurde die Bezeichnung „Turbo-Flash" gewählt, da das Bildmaterial von einem MRT der Firma Siemens stammt. Tabelle 3.3 zeigt die Bezeichnungen verschiedener Gerätehersteller (in wertungsfreier, alphabetischer Reihenfolge).

Bei dieser schnellsten MRT-Technik wird eine T_1-Wichtung durch einen 180°-Inversions-Präparations-Impuls aufgeprägt, die Gewinnung der Bilder beruht dann auf der Messung der Länge des Magnetisierungsvektors *M*. Fettgewebe mit einer kurzen Relaxationszeit stellt sich dabei im Bild heller, die Gewebe der Niere mit mittlerer Relaxationszeit etwas dunkler dar. Durch eine Eigenart der Aufnahmetechnik entsteht dabei an der Grenze zwischen Niere und umgebenden Gewebe eine scheinbare Konturbetonung. Abbildung 3.5 zeigt eine Vergrößerung der Nieren aus der in der Einleitung abgebildeten Serie. Auf die Konturbetonung wird in Kapitel 7.8 und Kapitel 9.10.1 noch zurückgegriffen.

Gerätehersteller	$GE_{gespoilt}$ T_1-betont	$GE_{refokussiert}$ T_1/T_2^*	$GE_{gespiegelt/refokussiert}$ T_2-betont
Diasonics	PFI		
Elscint	FE		
Fonar	Gradienten-Echo		
General Electric		GRASS	
Philips	FFE		
Picker	Partial Saturation Recovery	FAST	CE-FAST
Siemens	FLASH	FISP	PSIF

Tabelle 3.3: Gradientenverfahren. Akronyme einiger Gerätehersteller [LISS90].

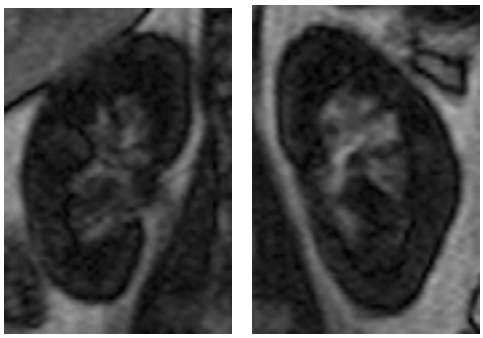


Abbildung 3.5: Konturbetonung an der Grenze zwischen Nieren und umgebendem Gewebe.

3.2.2 Bilddarstellung und Fensterung

Zur bildlichen Darstellung werden den bei der MRT gemessenen EM-Signalen verschiedene Grauwerte zugeordnet. Viele Bildverarbeitungssysteme können aber nur 256 Grauwerte darstellen, was aufgrund der Wahrnehmungsfähigkeiten des menschlichen Auges auch sinnvoll und völlig ausreichend ist, siehe [FELL92]. Da aber MRT-Bilder, je nach Art der Aufnahme, einen wensentlich größeren Grauwertbereich belegen können (bis zu 5000 verschiedene Werte), besteht die Möglichkeit der sogenannten Fensterung. Diese geschieht bei modernen Programmen meist automatisch und wird von Benutzer nicht wahrgenommen. Bei der interaktiven Fensterung kann der Benutzer Fensterbreite und –position zur Dastellung bestimmen. Dieses Fenster wird durch Angabe des Zentrums (engl. center) und der Breite (engl. window) definiert, siehe Abbildung 3.6. Durch Verschieben des Zentrums erreicht man eine Variation der Helligkeit des Bildes, durch Veränderung der Fensterbreite verändert man den Kontrast. Voraussetzung dafür ist natürlich, daß der ausgewählte Grauwertbereich auf die maximale Anzahl darstellbarer Grauwerte expandiert wird. Somit können anhand der gleichen MRT-Aufnahme mit verschiedenen Fenstern unterschiedliche medizinische Fragestellungen beantwortet werden.

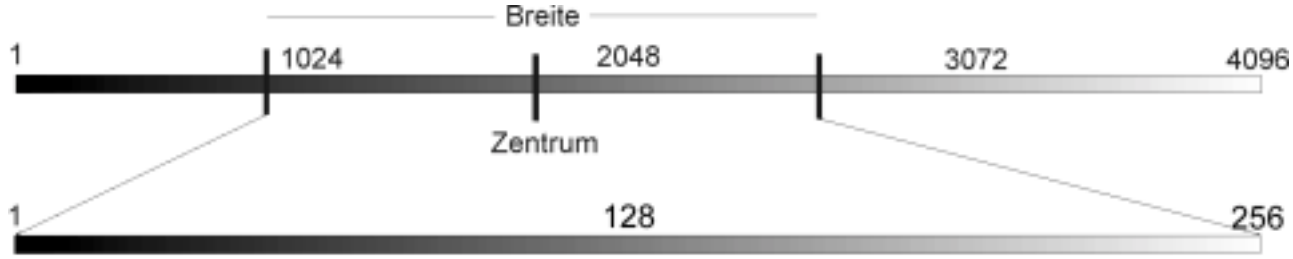


Abbildung 3.6: Fenstern.

4 Funktionale Beschreibung des Systems

Im folgenden soll kurz auf das im Rahmen dieser Arbeit erstellte Programm eingegangen werden. Dabei sollen die Wege zur Korrektur der Atmungsbewegung des Bildmaterials und zur Berechnung der Ergebnis-Meßkurven erklärt werden. Die dahinterstehenden Methoden und Ideen werden hier bewußt nur kurz beschrieben, so daß die Funktionsweise im Vordergrund steht. Eine ausführliche, theoretische Behandlung folgt dann in den nächsten drei Kapiteln. Intention dieser Gliederung ist es, besonders Lesern aus dem medizinischen Bereich eine anschauliche, praktische Einführung in die Funktionsweise des Systems zu geben, ohne in die Mathematik und Informatik der Bildverarbeitung einzutauchen. Der Leser sollte mit Abschluß dieses Kapitels einen kompletten Überblick über die Thematik und Lösungsmethoden bekommen haben.

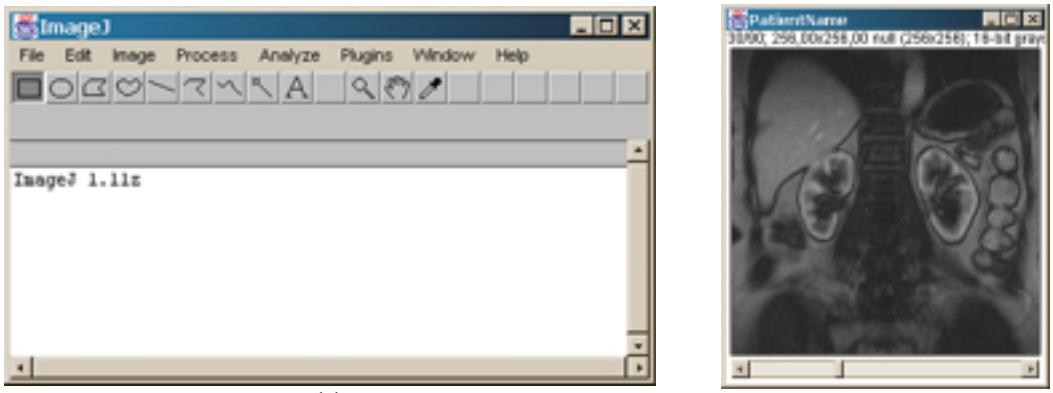
Anwender, d.h. Diagnostiker, sind nach diesem Kapitel bereits in der Lage, das System zu bedienen. Jedoch sollten sie zum besseren Verständnis weiterlesen, besonders Kapitel 8 über die Methodik zur Berechnung der Meßkurven sowie die Installations- und Bedienungshinweise in Kapitel 10 sind dabei unumgänglich.

4.1 Bildmaterial

Das der Untersuchung der funktionellen Nierenperfusion zugrundeliegende Bildmaterial besteht aus einer Serie von 90 Aufnahmen. Die Bilder werden mit einem 1,5 *Tesla* Siemens „Magnetom Vision“ mittels einer Turbo-Flash-Sequenz mit einem Akquisitionsintervall von ca. 2 Sekunden aufgenommen. Die reale Bildgröße (engl. field of view, FOV) beträgt 30×30 *cm* bei einer Auflösung von 256×256 Bildpunkten und einer Dicke von 10 *mm*. Die Sequenz zeigt eine leicht angulierte, koronare[6] Schicht des Abdomenbereichs, die so vom Benutzer gewählt wird, daß beide Nieren mit möglichst maximaler Fläche abgebildet sind. Zu Beginn der Aufnahme wird dem Patienten ein Kontrastmittel intravenös gegeben, i.d.R. Gd-DTPA[7] oder ein Analogon. Dieses flutet im zeitlichen Verlauf der Akquisition in die Niere ein und bewirkt in den verschiedenen Gewebearten der Niere einen unterschiedlich starken Anstieg der Signalintensität, deren Messung die Grundlage der späteren Diagnosestellung ist.

4.2 Bildverarbeitung mit IMAGEJ

Als Grundlage für die Entwicklung des Systems wurde das bestehende Bildverarbeitungsprogramm IMAGEJ gewählt [RASB98]. Viele Operatoren der Bildverarbeitung (siehe Kapitel 5) sind bereits darin enthalten und brauchten daher bei der Entwicklung nicht neu implementiert werden. Auch die Möglichkeit, Bilderserien als „Bilderstapel“ darzustellen und mit einem Schieberegler „durchzublättern“, ist bereits vorhanden. IMAGEJ ist in JAVA geschrieben, der Quellcode ist öffentlich verfügbar und die Möglichkeit der Erweiterung durch eigenständige Module (engl. *Plugins*) ist bereits im Konzept verankert. Es bildet damit eine gute, plattformunabhängige Basis und entspricht optimal den Anforderungen des in dieser Arbeit zu erstellenden Systems.

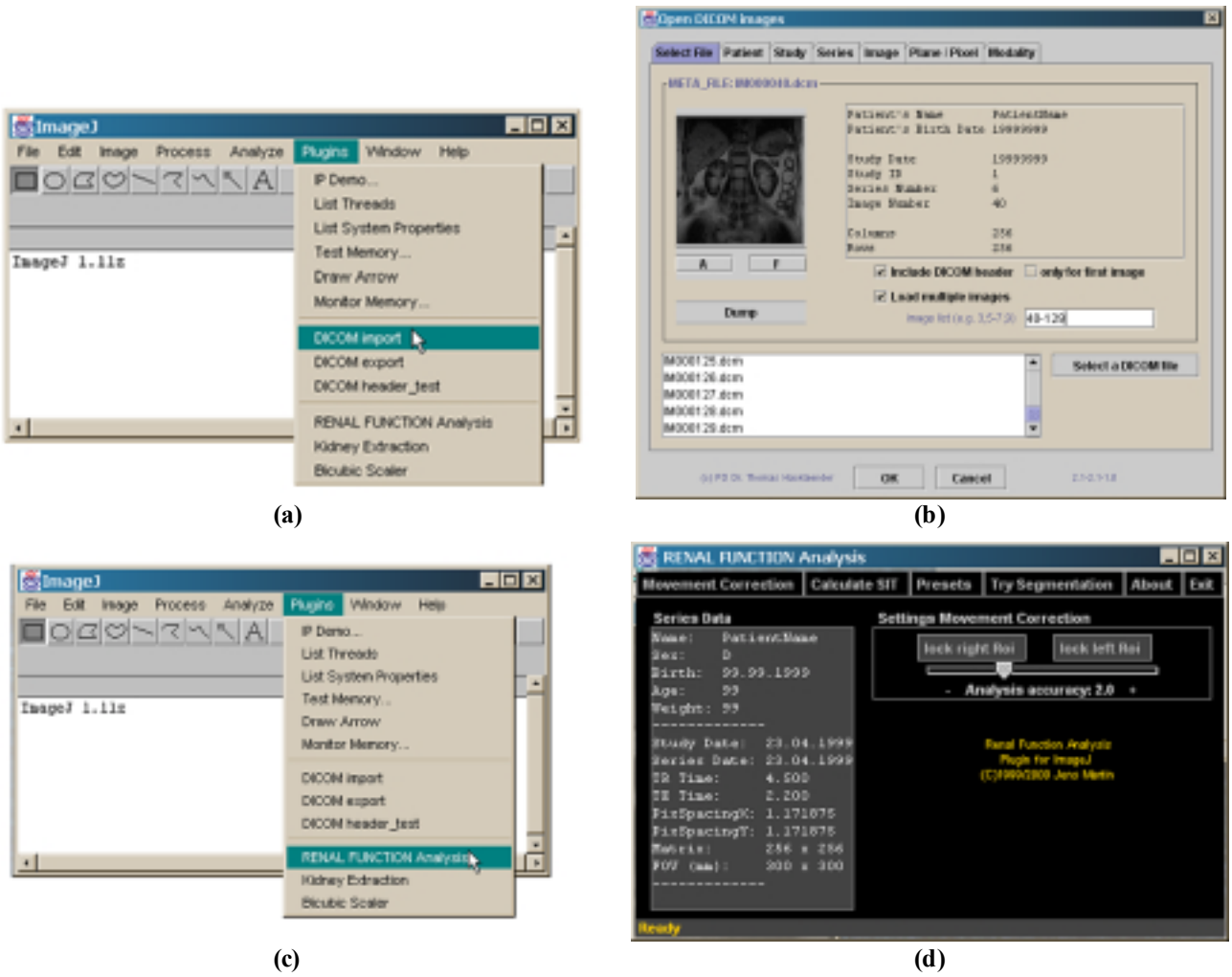


(a) **(b)**

Abbildung 4.1: IMAGEJ (a) und eine Nierensequenz als Bilderserie (b).

Desweiteren kann zum Import der Bilddaten auf ein bestehendes Modul zurückgegriffen werden [HACK98]. Dieses ist in der Lage, das medizinische DICOM-Format zu lesen und kann lokale oder über ein Netzwerk zugängliche Daten lesen. Sowohl das Import-Modul als auch das Modul zur renalen Funktionsanalyse sind in *Plugins* für IMAGEJ gekapselt und können jeweils über den Menüpunkt *Plugins* ⇨ *DICOM-Import* bzw. *Plugins* ⇨ *RENAL FUNCTION Analysis* aufgerufen werden, siehe Abbildung 4.2.

[6] koronar = längs zur Körperachse.
[7] Das paramagnetische Kontrastmittel Gadolinium-DPTA wirkt nicht nephrotoxisch und kann daher auch bei eingeschränkter Nierenfunktion benutzt werden.



(a) (b)

(c) (d)

Abbildung 4.2: Verwendete *Plugins* von IMAGEJ und deren Aufruf. Dicom-Import-*Plugin* (a) und (b) und das *Plugin* zur renalen Funktionsanalyse (c), (d).

4.3 Korrektur der Atmungsbewegung

Die Probleme bestehender computerunterstützter Untersuchungsverfahren zur Nierenperfusion [SCHU91], [MILL99] resultieren häufig aus der atmungsbedingten Verschiebung des Organs während der Bildgewinnung und der starken inhaltlichen Varianz der Aufnahmen.

Die Informatik bezeichnet mit dem Begriff Bildregistrierung u.a. das Problem, ein Referenzbild und ein Folgebild mit gleichem, aber verschobenem Inhalt so zu justieren, daß die abgebildeten Objekte deckungsgleich werden. Genau dieser Sachverhalt stellt sich bei den Nierensequenzen dar und kann daher mit einer Methode aus dem Bereich der Bildregistrierung bearbeitet werden. Die im nächsten Abschnitt vorgestellte Kepstrum-Filterung bietet dabei einige Vorzüge und kommt daher in diesem System zur Anwendung. In Bezug auf die Nierensequenzen muß noch gesagt werden, daß die Verarbeitung für die linke und rechte Niere getrennt durchgeführt wird, indem die Sequenz einfach in der Mitte vertikal aufgeteilt wird. Die Anatomie des menschlichen Bauchraumes und der Organe darin ist – mit Ausnahme der beidseitig gelegenen Nieren – weitgehend asymmetrisch, weswegen sich für die linke und rechte Niere meist unterschiedlich starke Atmungsverschiebungen ergeben.

4.3.1 Kepstrum-Filter

Zur Bildregistrierung existiert inzwischen eine Vielzahl von Methoden, von denen einige am Anfang des Kapitels 6 beschrieben werden. Die klassische, zweidimensionale Kepstrum-Technik, ein der Fourier-Analyse zuzuordnendes Verfahren, eignet sich sehr gut zur Detektion von verschobenen

Bildinhalten (Translationen) und hat sich als äußerst robust und rauschunempfindlich erwiesen. Sie reagiert zwar empfindlich auf Rotationen [GOER95], jedoch ist die Atmungsverschiebung der Nieren in allen vorliegenden Sequenzen in etwa linear. Minimale Rotationen liegen alle im Toleranzbereich des Kepstrum-Filters (±4° Drehwinkel). Ein weiterer Vorteil dieses Verfahrens ist, daß keine zusätzlichen Annahmen über den Bildinhalt (Objektform, etc.) benötigt werden.

Andere Verfahren, wie z.B. die Segmentierung, erwarten häufig feste Bildregionen, Intensitätsbereiche oder vorgegebene Objektkonturen. Durch die variable Anatomie der Natur sowie teilweise vorliegende Organschäden variieren die Nierensequenzen aber stark in ihrem Bildinhalt und erfüllen daher die zuvor genannten Voraussetzungen nur ungenügend.

Zur Durchführung der Kepstrum-Filterung werden das Referenzbild und das verschobene Folgebild direkt nebeneinander mittig in das sogenannte Kepstrum-Fenster kopiert. Abbildung 4.3 zeigt ein Beispiel. Nun wird die Filterung durchgeführt, wobei der Hauptbestandteil der Berechnung aus zwei Fourier-Transformationen besteht. Als Ergebnis erhält man ein Bild im Kepstral-Bereich, die sogenannte Kepstral-Ebene, in der die Verschiebung des Folgebildes gegenüber dem Referenzbild kodiert ist (siehe Abbildung 4.4). Da die Ausmaße der beiden Eingabebilder die Größe des Kepstrum-Fensters bestimmen, kann ein Suchbereich festgelegt werden, der durch den gestrichelten Bereich in der Abbildung der Kepstral-Ebene angedeutet wird. Die Verschiebung kann nun durch eine einfache Detektion des Bereichs maximaler Intensität (engl. peak) und dessen Abstand vom Mittelpunkt (Ursprung) des Suchbereichs bestimmt werden, wie in Abbildung 4.4 dargestellt. Anschließend wird das Folgebild entsprechend der berechneten Translation justiert, danach liegt die Niere in beiden Bildern deckungsgleich an identischer Position.

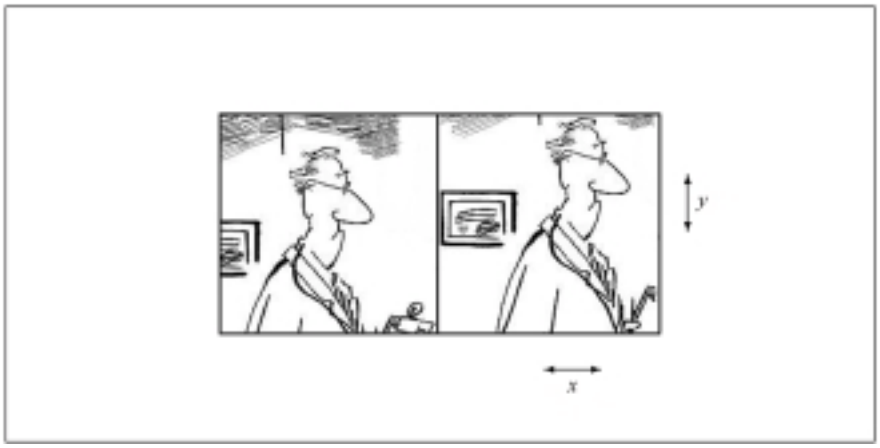


Abbildung 4.3: Anordnung von Referenzbild und Folgebild im Keptrum-Fenster.

In der Realität sind einige Vorverarbeitungsschritte notwendig [LEE89], die jedoch ohne Verständnis einiger Bildverarbeitungs-Operatoren schwer zu erläutern sind und daher hier nicht weiter diskutiert werden. Es sei aber noch bemerkt, daß aufgrund der relativ niedrigen Auflösung von 256×256 Bildpunkten häufig der Fall auftritt, daß die *x*- oder *y*-Verschiebung einen nicht-ganzzahligen Betrag hat. Daher muß das System in der Lage sein, auch Bilder mit einer Verschiebung der Niere von beispielsweise x=14,5 / y=8,5 möglichst gut zu korrigieren. Um die Genauigkeit der Verschiebungsberechnung zu erhöhen, können daher die Bilder vor der Kepstrum-Filterung um einen beliebigen Faktor $m \in \mathbb{R}$ vergrößert werden, anschließend werden dann die Berechnungen durchgeführt und die errechnete Translation zum Schluß wieder durch m geteilt.

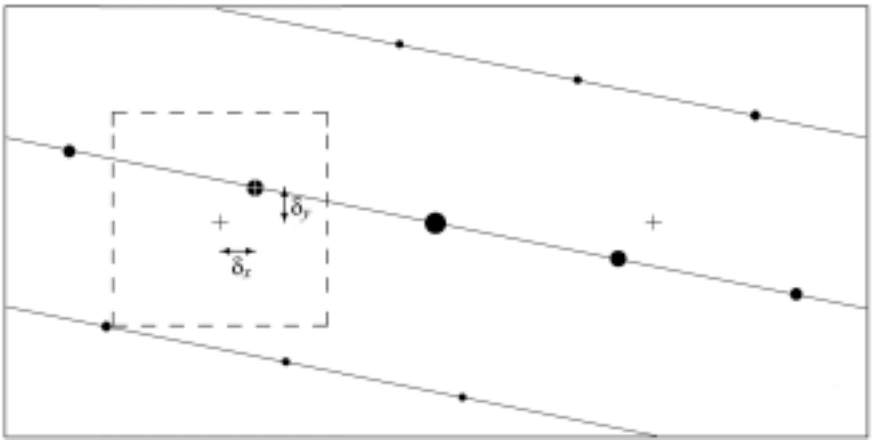


Abbildung 4.4: Peaks in der Kepstral-Ebene.

4.3.2 Algorithmus zur Bewegungskorrektur von Nierenserien

Der Algorithmus zur Translationskorrektur der Nieren-Serien vergleicht sequentiell jeweils zwei Bilder n und $n+d$ der Serie, wobei n das Referenzbild und $n+d$ das Folgebild sei. Zu Beginn gilt $n=d=1$. Anschließend wird die Verschiebung des Folgebildes korrigiert und der Algorithmus rückt eine Position weiter (erhöhe n, $d=1$). Diese Art der Bearbeitung der Nierensequenzen hat das Problem, das ein gerade justiertes Folgebild im „nächsten Durchlauf" zu einem Referenzbild wird und die Position der Niere als korrekt vorausgesetzt wird. Tritt nun bei der Kepstrum-Filterung ein Fehler[8] auf und ist das Folgebild des letzten Durchlaufs falsch justiert, so wird es zu einem falschen Referenzbild. Zur Vermeidung derartiger sich fortsetzender Fehler muß zur Laufzeit eine qualitative Ergebniskontrolle durchgeführt werden. Form und Ausprägung des Maximums in der Kepstral-Ebene stellen beispielsweise geeignete Bewertungskriterien dar. Mißlingt die Translationskorrektur, so können einzelne Bilder als ungültig markiert werden und bleiben in der nächsten Iteration als (falsches) Referenzbild unberücksichtigt. Es wird dann lediglich d erhöht. Abbildung 4.5 veranschaulicht den Arbeitsablauf anhand eines Flußdiagramms.

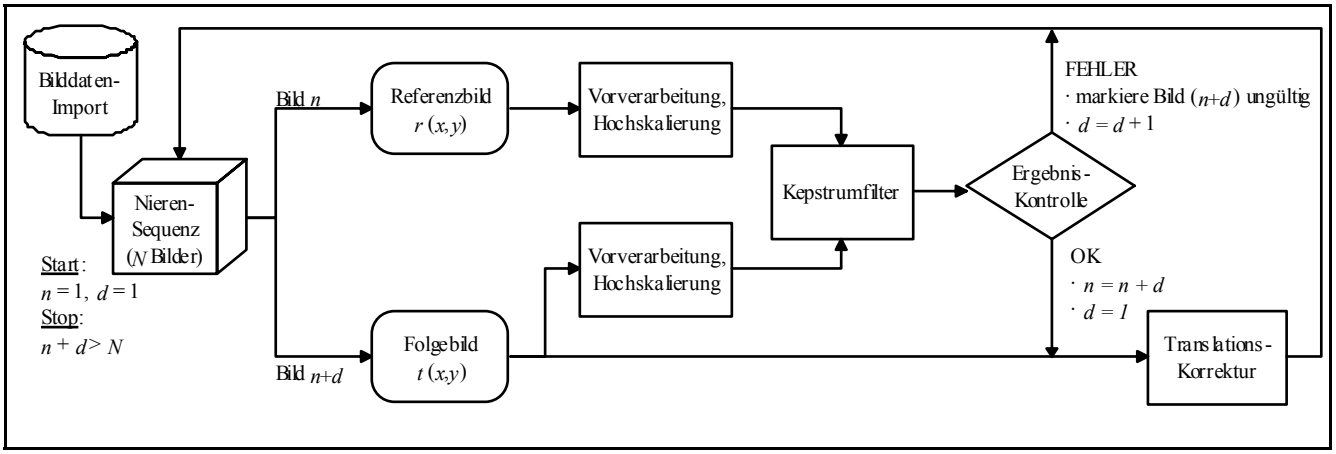


Abbildung 4.5: Arbeitsweise des Algorithmus zur Translationskorrektur.

4.4 Berechnung der Meßkurven

Um medizinisch relevante Aussagen treffen zu können, muß das Kontrastmittelverhalten für das Gesamtorgan und verschiedene Regionen der Niere betrachtet werden [KNOO97], [MILL99]. Grundlage für eine einfache Auswertung bilden die bewegungskorrigierten Bilddaten. Unter der

[8] Die Ursachen für mögliche Fehler sind meist ein mangelhaftes Bildmaterial, siehe auch Kapitel 7.7

Annahme, daß die Niere in allen Bildern der Sequenz deckungsgleich an identischer Position liegt, muß der Diagnostiker nur in einem einzigen beliebigen Serienbild einmal eine ROI entlang der Kontur des Gesamtorgans einzeichnen. Für alle Bildpunkte innerhalb der Markierung wird dann z.B. die mittlere Intensität berechnet. Durch Projektion dieser ROI auf alle anderen Bilder der Serie errechnet sich die gesuchte Kurve für die Gesamtniere, siehe Abbildung 4.6. Diagnostisch interessant ist dabei der Vergleich zwischen linker und rechter Niere: Jede einseitige Verzögerung oder Abflachung deutet auf eine vorliegende Nierenschädigung oder Nierenarterienstenose hin.

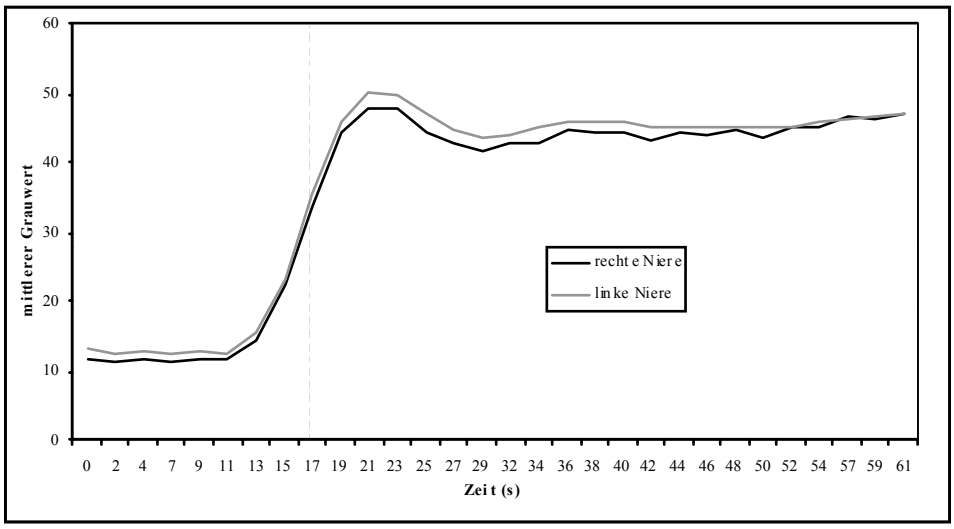


Abbildung 4.6: Meßkurve einer Beispielsequenz für die gesamte Niere (Bilder 1-30).

4.4.1 Differenzierung der Nierengewebe

Da auch Signal-Intensitäts-Kurven für die funktionell unterschiedlichen Anteile der Niere, nämlich Nierenrinde und -mark berechnet werden sollen, müssen die Bildpunkte der ROI für die gesamte Niere einer dieser beiden Gewebegruppen zugeordnet werden. Dabei kann a-priori-Wissen über den Kurvenverlauf für das Gesamtorgan ausgenutzt werden, denn selbst bei stark geschädigten Nieren nimmt zunächst die Nierenrinde Kontrastmittel auf. Der Anstieg der Kurve in Abbildung 4.6 wird also durch hellere Intensitäten der Bildpunkte der Rinde bewirkt, erst mit zeitlich deutlicher Verzögerung nimmt auch das Nierenmark KM auf. Eine maximale Anreicherung in beiden Gewebearten bewirkt schließlich den Peak bei 1/3 der Meßzeit.

Betrachtet man nun die Kurve als eine Funktion der Zeit, so kann durch eine Glättung und Bildung der zweiten Ableitung der Wendepunkt bestimmt werden, wie im Bild durch die rote Linie dargestellt. Dessen Betrag repräsentiert nun genau die Bildnummer der Sequenz, in der zur Rinde gehörenden Bildpunkte schon KM aufgenommen haben, Bildpunkte des Marks hingegen noch nicht. Der Bereich des Nierenbeckens ist für die Untersuchungsergebnisse uninteressant und kann, da seine Bildpunkte über die gesamte Serie gar kein oder nur sehr wenig Kontrastmittel aufnehmen, ebenfalls einfach eingegrenzt werden.

Basierend auf den genannten Eigenschaften können nun Regeln für ein mathematisches Verfahren erstellt werden, welches die Bildpunkte entsprechend ihren Eigenschaften den Gewebegruppen zuordnet. Etwas vereinfacht kann man sich z.B. Grenzwerte für maximale oder minimale Intensität vorstellen. Abbildung 4.7 zeigt das Resultat der Klassifizierung einer Beispielsequenz.

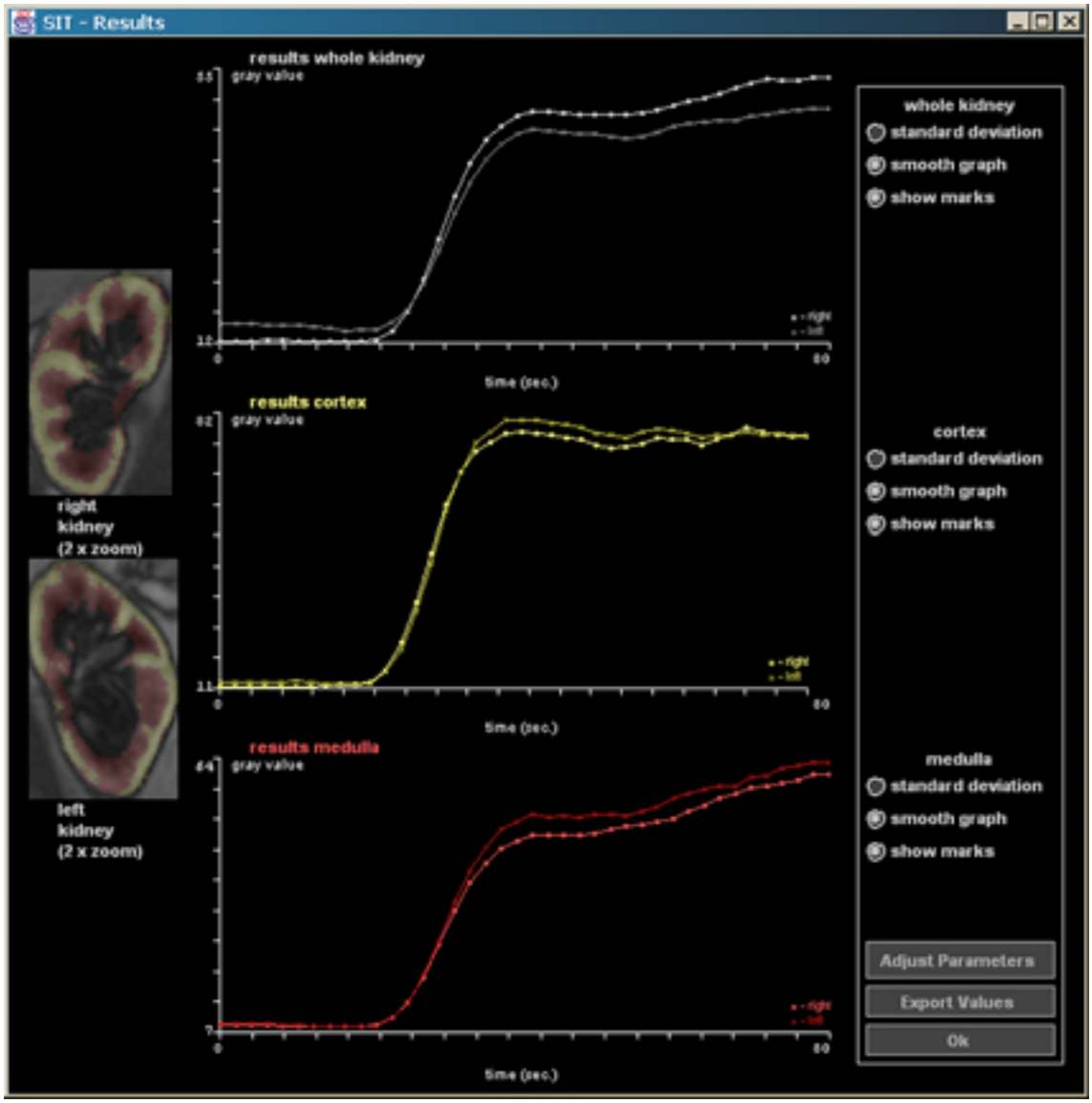


Abbildung 4.7: Automatisch berechnete Meßkurven für eine Beispielsequenz (Bilder 1-40).

4.5 Manuelle Auswertung

Sollte die Bewegungskorrektur aufgrund schlechter Einstellungen oder ungenauer Aufnahmen einmal nicht die gewünschten Ergebnisse liefern oder der Diagnostiker mit der Klassifizierung der verschiedenen Gewebe nicht einverstanden ist, so bietet das System die Möglichkeit, eine (konventionelle) manuelle Auswertung durchzuführen. Zu diesem Zweck können in jedes Halbbild der Serie eine ROI für das Gesamtorgan und jeweils bis zu 100 ROIs für Nierenrinde und -mark getrennt eingezeichnet werden. Abbildung 4.8 zeigt ein (vergrößertes) Beispiel, dabei ist die Gesamt-ROI grün markiert, die ROIs für die Nierenrinde gelb und für das Mark rot. Die Berechnung der Meßkurven erfolgt dann anhand der eingezeichneten ROIs. Die Berechnungsschritte zur Kepstrum-Filterung und zur Gewebeklassifizierung sind dann selbstverständlich nicht notwendig.

Abbildung 4.8: Manuelles Einzeichnen multipler ROIs.

4.6 Export der Ergebnisse

Zur Weiterverarbeitung mit anderen Programmen können die berechneten Ergebnisse exportiert werden, z.B. zur Berechnung von Kurvenparametern, die die Nierenperfusion beschreiben. Als Format für den Datenexport wurde ein Textformat mit Tabulatoren als Trennzeichen verwendet, da dies von allen Tabellenkalkulations-Programmen oder Datenbank-Programmen gelesen werden kann. Die exportierte Textdatei enthält zunächst einen Kopf (engl. header) mit Informationen zum Patienten und zur Aufnahme, sowie tabellarisch die Meßwerte für die rechte und linke Gesamtniere, das Nierenmark und die Nierenrinde.

4.7 Übersicht

Durch die in diesem Kapitel vorgestellten Komponenten und den beschriebenen Bearbeitungsweg läßt sich das System zur renalen Funktionsanalyse durch das in Abbildung 4.9 gezeigte Diagramm veranschaulichen. Da das Diagramm in Kapitel 9 nochmals verwendet wird, ist der Begriff *ImagePlus* zum jetzigen Zeitpunkt noch unklar. In einem *ImagePlus*-Objekt speichert IMAGEJ den kompletten Bilderstapel (Nierenserie) sowie einge weitere Daten zur Bildsequenz und Funktionen zur Bildbearbeitung.

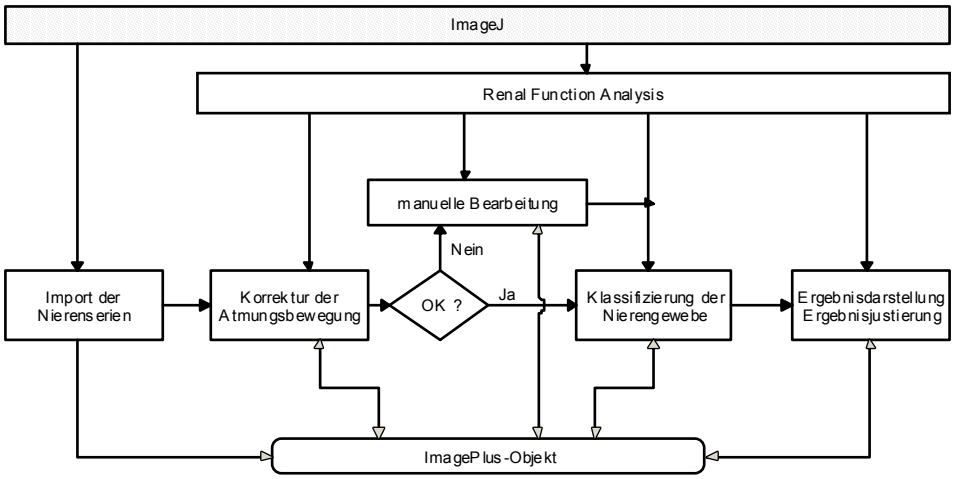


Abbildung 4.9: Übersicht über die Komponenten des Systems sowie den Bearbeitungsweg.

5 Grundlagen der Bilddarstellung und Bildverarbeitung

In diesem Kapitel werden die Grundlagen für die verschiedenen Darstellungsformen von Bildern im Computer behandelt. Während im bisherigen Verlauf der Arbeit bewußt auf den Einsatz von mathematischen Definitionen, Formeln und Sätzen verzichtet wurde, so müssen für die Lektüre der folgenden drei Kapitel Grundkenntnisse der Informatik und insbesondere der Mathematik vorausgesetzt werden. Ohne eine Einführung in die Bildverarbeitung lassen sich die Sachverhalte der Kepstrum-Filterung zur Korrektur der Atmungsbewegung nur unzureichend erklären. Aus den gleichen Gründen bedarf es auch einer Einführung in die Fourier-Transformation und deren Anwendung im Rahmen der Bildverarbeitung.

Da die digitale Bildverarbeitung ein sehr großes – und ständig wachsendes – Gebiet der Informatik ist, beschränken sich die folgenden Beschreibungen auf die unverzichtbaren Grundlagen und die im weiteren Verlauf angewendeten Operatoren. Alle dabei verwendeten Schreibweisen sind möglichst konsistent gewählt und werden im weiteren Verlauf der Arbeit beibehalten

Ein gewisser Lehrbuchcharakter läßt sich diesem Kapitel sicherlich nicht absprechen. Nachfolgend wird deshalb anhand von Abbildung 5.1 die Anwendung verschiedener Operatoren demonstriert. Indem ein konstantes Beispielbild zur Veranschaulichung gewählt wird, lassen sich die Ergebnisse auch besser vergleichen.

Abbildung 5.1: Beispielbild zur Veranschaulichung verschiedener Bildverarbeitungsoperatoren.

5.1 Bilder

Ein digitales Bild f der Dimension $M \times N$ wird mathematisch als Funktion betrachtet, die jedes Pixel auf einen Wertebereich, z.B. auf die Grauwertmenge G abbildet:

$$f : M \times N \rightarrow G \qquad (5.1)$$

Das kleinste Element einer Menge ist, wenn nicht anders angegeben, die Null. Wenn nun für die Zielmenge im Beispiel $|G|=256$ (Grauwerte) angenommen werden, gibt es in der Menge die Elemente $g \in G = \{0,1,...,g_{\max}\} = \{0,1,...,255\}$, mit $g_{\max} = |G| - 1$. Somit läßt sich (5.1) auch schreiben als

$$f(m,n) = \begin{cases} g & \text{für } m = \{0,...,m_{max}\},\ n = \{0,...,n_{\max}\} \\ 0 & \text{sonst} \end{cases} \qquad (5.2)$$

mit $m_{\max} = |M| - 1$, $n_{\max} = |N| - 1$

Der allgemein gebräuchliche Begriff Pixel für Bildpunkt (vom engl. **pi**cture **el**ement) entspricht dann dem Funktionswert an den (Bild-) Koordinaten (m,n), die durch den Pixel festgelegt werden. Für digitale (oder auch diskrete) Bilder gilt $m,n \in \mathbb{Z}$ $\mathbb{Z}$ oder $m,n \in \mathbb{N}$ (Anm.: Fast alle Computerprogramme zur Bearbeitung von Bildern stellen die Bilddaten in einem Koordinatensystem mit dem Ursprung (0,0) in der linken oberen Ecke des Bildes dar). Man spricht dann auch vom diskreten Bereich oder Ortsbereich. Die Darstellung kontinuierlicher Bilder $f(x,y)$ mit $x,y \in \mathbb{R}$ oder $x,y \in \mathbb{C}$ im Frequenzbereich wird in Kapitel 5.3 genauer betrachtet. Bei den in der Medizin zunehmend vorkommenden dreidimensionalen Bilddaten werden Bildfunktion und Pixel auf drei Parameter ausgeweitet. Man spricht dann nicht mehr von einem Pixel, sondern von einem Voxel (von **Vo**lumenpi**xel**).

Das dieser Arbeit zugrundeliegende Bildmaterial beschränkt sich jedoch auf den zweidimensionalen Bereich, weswegen der Leser häufiger mit dem Begriff Pixel konfrontiert wird. Die analysierten Bilder sind dabei ausschließlich Grauwertbilder, so daß ein eindimensionaler Wertebereich für Bildfunktionen angenommen werden kann. Lediglich zur Visualisierung der Klassifizierungsergebnisse (siehe Kapitel 8) werden Farbbilder verwendet, deren Wertemenge dreidimensional ist. Die Bildfunktion liefert dann für einen Pixel einen Funktionswert, der sich aus drei Intensitätswerten der Grundfarben (Rot, Grün, Blau) zusammensetzt: $f(m,n) = (r,g,b)$.

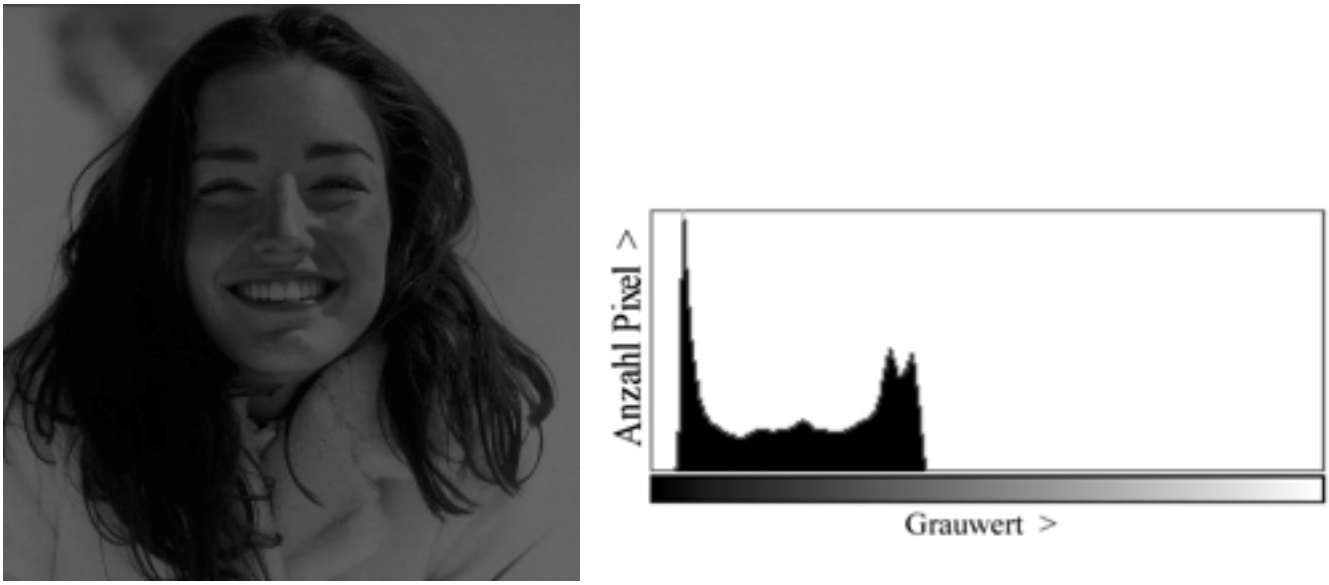
Die Mischung der drei Grundfarben an diesem Bildpunkt bestimmt dann die wahrgenommene Farbe.

5.1.1 Bildeigenschaften

Das **Histogramm** stellt die Grauwertverteilung oder Farbverteilung der Pixel eines Bildes dar, siehe Abbildung 5.2. Da, wie eingangs erwähnt, das Bildmaterial dieser Arbeit ausschließlich Grauwertbilder beinhaltet, wird im folgenden *immer* von einem eindimensionalen Wertebereich der Bilder ausgegangen. Die Histogrammfunktion eines Grauwertbildes ist definiert als:

$$h(i) = \sum_{m=0}^{|M|-1} \sum_{n=0}^{|N|-1} \varphi(f(m,n) - i) \qquad mit \quad \varphi(\xi) = \begin{cases} 1 & \text{für } \xi = 0 \\ 0 & \text{sonst} \end{cases} \qquad (5.3)$$

Allein durch Betrachtung des Histogramms lassen sich häufig schon Aussagen über die Bildqualität machen. So ist das Histogramm eines sehr dunklen und kontrastarmen Bildes schmal und hat eine deutliche Spitze (Peak) im unteren Grauwertbereich, siehe auch Abbildung 5.2b. Ein in etwa gleichverteiltes Histogramm ergibt sich in der Regel, wenn sowohl die Helligkeit als auch der Kontrast eines Bildes subjektiv optimal sind (z.B bei einer gut ausgelichteten, kontrastreichen Aufnahme eines Wohnraumes).



(a) (b)

Abbildung 5.2: Porträtaufnahme (zu dunkel) mit zugehörigem Histogramm.

Rückschlüsse auf den Bildinhalt sind ebenfalls möglich. Sind in einem Bild z.B. 3 große Gegenstände unterschiedlicher Helligkeit dargestellt, so zeigt das zugehörige Histogramm 3 deutliche Peaks.

Der **mittlere Grauwert** g_{av} ist eine weiterere Größe zur Beurteilung von Grauwertbildern. Er berechnet sich aus allen Pixeln eines Bildes und ist definiert als:

$$g_{av}(f) = \frac{1}{|M| \cdot |N|} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) \qquad (5.4)$$

Die **Standardabweichung** g_{dev} eines Grauwertbildes läßt sich erst mit Hilfe des mittleren Grauwertes berechnen. Sie läßt sich durch die folgende Gleichung darstellen:

$$g_{dev}(f) = \frac{1}{|M| \cdot |N|} \sum_{m=0}^{|M|-1} \sum_{n=0}^{|N|-1} \left(f(m,n) - g_{av}(f)\right)^2 \qquad (5.5)$$

Die Standardabweichung ist somit ein globales Maß der Abweichung aller Pixel vom mittleren Grauwert g_{av} eines Bildes. Bilder, die eine hohe Standardabweichung haben, sind sehr kontrastreich.

5.1.2 Modifikation der Grauwertverteilung

Oft ist es nötig, die Grauwertverteilung eines Bildes zu manipulieren, z.B. um den Kontrast zu verbessern oder die Helligkeit eines Bildes einem anderen Bild (Referenzbild) anzupassen. Zu diesem Zweck definiert man eine Abbildung oder Transformation $\mathcal{T}$, die ein Bild f auf ein transformiertes Bild f' abbildet:

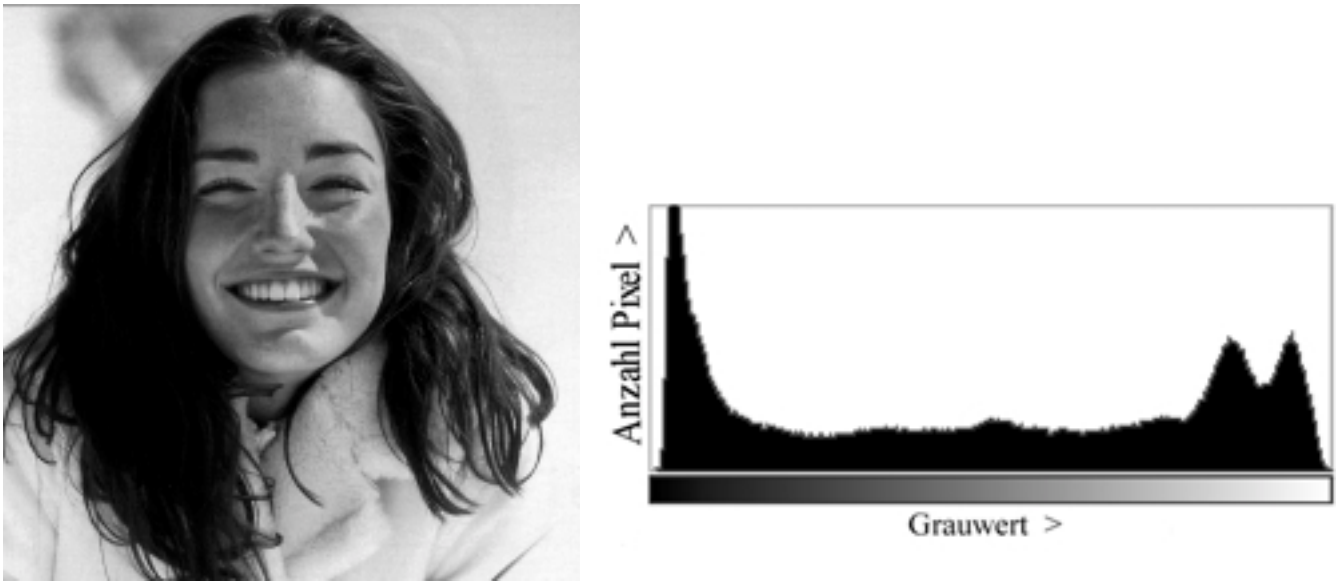
$$\mathsf{T} : f \to f' \quad \text{oder äquivalent:} \quad \mathsf{T}(f) = f' \qquad (5.6)$$

Soll eine lineare Skalierung der Grauwertverteilung erreicht werden, so kann dies z.B. durch eine lineare Abbildung geschehen:

$$f'(m,n) = \left(f(m,n) + c_L\right) \cdot c_C = c_C \cdot f(m,n) + c_L \cdot c_C \qquad (5.7)$$

Durch die beiden Parameter c_L und c_C können dabei Helligkeit und Kontrast gesteuert werden. Für $c_L = 0$ und $c_C = 1$ ergibt sich die identische Abbildung. Ist $c_L > 0$, so wird zu den Grauwerten der Ausgangsbildes eine Konstante addiert und das Bild wird insgesamt heller, während für $c_L < 0$ das Bild dunkler wird. Das zu f' gehörige Histogramm ist gegenüber dem von f nach links $(c_L < 0)$ oder rechts

$(c_L > 0)$ verschoben. Der Parameter c_C bewirkt eine Änderung des Kontrastes. Für $|c_C| > 1$ wird das Histogramm breiter und das Bild kontrastreicher, $|c_C| < 1$ macht das Histogramm schmaler und das Bild kontrastärmer. Bei der Transformation nach dem hier angegebenen Schema muß noch beachtet werden, daß unter Umständen der Wertebereich G' des transformierten Bildes f' größer als der ursprüngliche Wertebereich G ist und außerhalb der definierten Schranken liegt (z.B. mehr als 256 Einträge). Entsprechend muß man entweder die Parameter c_L und c_C in (5.7) so anpassen, daß $G' = G$ gilt oder zu einer stückweise linearen Skalierung übergehen. Da das im Rahmen dieser Untersuchung verwendete Bildmaterial ausreichend kontrastiert ist, soll hier nicht weiter darauf eingegangen werden. Ausführliche Anleitungen zu Bildverbesserung findet der Leser in [GONZ92], [HABE95] und [WAHL89].



(a) **(b)**

Abbildung 5.3: Siehe Abbildung 5.2a, mit modifizierter Grauwertverteilung.

5.1.3 Differenzbilder

Der Begriff des Differenzbildes wird im weiteren Verlauf der Arbeit häufig benötigt. Das Differenzbild f_{diff} zweier Bilder f_1 und f_2 ist definiert als

$$f_{diff}(m,n) = \left| f_1(m,n) - f_2(m,n) \right| \qquad (5.8)$$

und kann somit als Aussagekriterium über die Gleichheit zweier Bilder herangezogen werden. Das Differenzbild zweier identischer Bilder ist an jeder Position (m,n) Null. Enthalten zwei Bilder im wesentlichen den gleichen Inhalt, aber unterscheiden sich in wenigen Strukturen, so werden genau diese im Differenzbild sichtbar. Abbildung 5.4 veranschaulicht dies an einem Beispiel. Später werden insbesondere der mittlere Grauwert g_{av} und die Standardabweichung g_{dev} des Differenzbildes von Interesse sein. Sie lassen sich leicht und schnell berechnen und können als numerische Aussage über die Gleichheit zweier Bilder herangezogen werden (vergleiche Kapitel 6.1).

(a) (b) (c)

Abbildung 5.4: Beispiel zur Bildung des Differenzbildes (c) durch Subtraktion des Bildes (b) von (a).

5.2 Ortsraum

Als Ortsraum wird im allgemeinen die diskrete Darstellung eines (digitalen) Bildes bezeichnet. So beziehen sich die in den beiden vorigen Abschnitten diskutierten Operatoren auf den Ortsraum.

5.2.1 Lokale Operatoren im Ortsraum

Lokale Operatoren im Ortsbereich (oder Ortsraum) bilden einen Intensitätswert g an der Stelle $f(m,n)$ innerhalb eines lokalen Bildbereiches $W_{m,n}$ auf einen neuen Wert g' ab. Abbildung 5.5 veranschaulicht diesen Vorgang. Für das Operatorfenster $W_{m,n}$ wird dabei ein Funktionalzusammenhang Θ definiert, für den gilt:

$$g' = f'(m,n) = \Theta[f(i,j)] \quad mit \quad (i,j) \in W_{m,n} \tag{5.9}$$

Ein sehr gebräuchlicher Funktionalzusammenhang ist z.B. das einfache Addieren der Bildwerte innerhalb des Operatorfensters, wobei (zusätzlich) die einzelnen Werte mit verschiedenen Gewichtungen versehen werden können. Zur Berechnung des Ergebnisbildes wird das Operatorfenster nun pixelweise über das Eingangsbild geschoben und jeweils Θ berechnet. Betrachtet man das Operatorfenster selbst als diskretes Bild und die Gewichtungen als Intensitäten an den (Pixel-) Koordinaten, kann man die durchgeführte Berechnung mathematisch als Faltung betrachten, wobei das Originalbild mit einem (kleineren) Bild gefaltet wird, dem sog. Faltungskern κ. Bei linearen Operatoren im Ortsbereich gibt man deswegen häufig die (Gewichtungs-) Matrix des Faltungskerns an, um die Arbeitsweise eines Filters zu verdeutlichen. Der im nächsten Abschnitt vorgestellte lineare Tiefpaßfilter besitzt z.B. den Faltungskern:

$$\kappa = \begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{5.10}$$

Die Form des Operatorfensters wird meist intuitiv festgelegt, am gebräuchlichsten sind aber rechteckförmige

$$W(m,n) = \{(i,j): m - \Delta m \le i < m + \Delta m \text{ und } n - \Delta n \le j < n + \Delta n\} \tag{5.11}$$

oder kreuzförmige symmetrische Einzugsbereiche. Die Größe Δm, Δn des Operatorfensters beeinflußt dabei das Ergebnis und die Rechenzeit. Je größer das Fenster, desto mehr Operationen müssen für die Berechnung von g' durchgeführt werden. Im Beispiel sind für jede Bildposition 17 Operationen nötig: 8 Summenbildungen und 9 Multiplikationen.

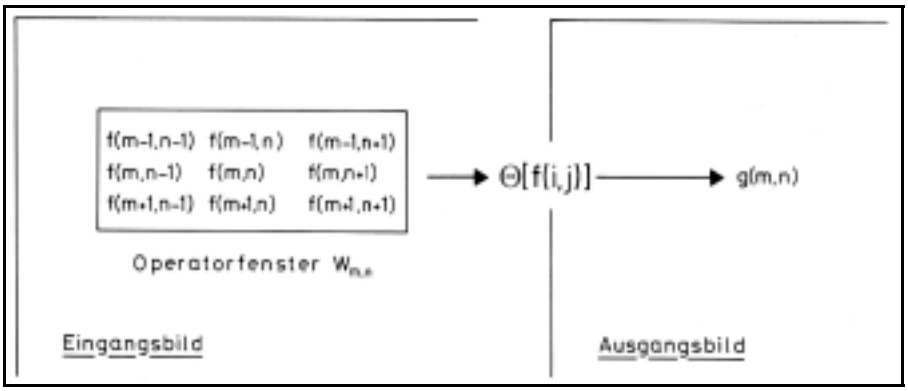


Abbildung 5.5: Prinzip lokaler Operatoren im Ortsraum.

5.2.2 Lineare Glättungsoperatoren

Lineare Glättungsoperatoren im Ortsbereich kommen immer dann zum Einsatz, wenn Bilddaten mit breitbandigen Störsignalen (Rauschen) überlagert sind, die häufig schon bei der Bildgewinnung entstehen. Der dem Leser vielleicht geläufigere Begriff „Tiefpaßfilter“ beschreibt die Arbeitsweise dieser Operatoren: Die das störende Rauschen verursachenden hohen Frequenzen im Bild werden herausgefiltert, tiefe Frequenzen dürfen passieren. Im Ortsraum wird dies durch eine einfache lokale Mittelung der Intensitätswerte innerhalb des Operatorfensters realisiert. Um eine Überschreitung des Ausgangswertebereiches G für das Ergebnisbild zu vermeiden, wird der berechnete neue Intensitätswert mit der Summe der Gewichte des Operatorfensters, z.B. für $\Delta m = \Delta n = 1$ mit $\frac{1}{9}$ multipliziert, siehe (5.10). Da sich aber nicht nur das Rauschen durch hochfrequente Bildsignale darstellt, sondern auch feine Bilddetails (z.B dünne Linien und Punkte, Kanten; aber auch Hell-Dunkel-Konturen) hohe Frequenzen im Bild verursachen, werden auch diese bei der Tiefpaßfilterung gedämpft und erscheinen im Ergebnisbild unscharf, siehe Abbildung 5.6b.

5.2.3 Median-Filter

Der Median-Filter vermeidet weitgehend diesen Nachteil der Tiefpaßfilterung. Beim Median-Filter werden die p Elemente (p sei der Einfachheit halber als ungerade Zahl angenommen) innerhalb das Operatorfensters der Größe nach geordnet. Das Operatorergebnis ist dann der $(p+1)/2$ -größte Wert dieser Reihe („Median“). Durch dieses Verfahren werden nur Strukturen beseitigt, die innerhalb des Operatorfensters nicht mehr als $(p-1)/2$ Bildpunkte betragen. Dies werden im allgemeinen nur Punkte, Linien und Kanten sein, Hell-Dunkel-Konturen bleiben hingegen erhalten und werden im Gegensatz zum vorher behandelten Tiefpaßfilter nicht unschärfer, siehe Abbildung 5.6c.

(a) **(b)** **(c)**

Abbildung 5.6: gestörtes Bild (a), mit lokalem Tiefpaßfilter geglättet (b), mit Median-Filter geglättet (c).

5.2.4 Lokale Gradientenoperatoren

Lokale Gradientenoperatoren sind spezielle Realisierungen der lokalen Operatoren. Mit ihrer Hilfe werden lokale Intensitätsdifferenzen (meist 1. oder 2. Ordnung) in Bildern ermittelt. Diese nehmen insbesondere in Kantenbereichen, d.h. in Grenzgebieten verschiedener im Bild dargestellter Objekte mit unterschiedlichen Intensitätswerten, hohe Werte an. Anhand des in Abbildung 5.7 gezeigten Diagramms lassen sich die meisten derartigen Verfahren gut veranschaulichen. In mehreren (meist zwei oder vier) parallelen Zweigen werden zunächst lokale Mittelwerte in einer Richtung berechnet. Im nächsten Schritt werden dann in dazu orthogonaler Richtung lokale Differenzwerte gebildet. Anschließend werden die Beträge der Zweige durch eine Summen- oder Maximumoperation zu einem Betragsgradientenbild $g(m,n)$ verrechnet. Manchmal wird auch nur ein Richtungsgradientenbild τ(m,n) benötigt, dann reicht i.d.R. die Berechnung eines Zweiges aus.

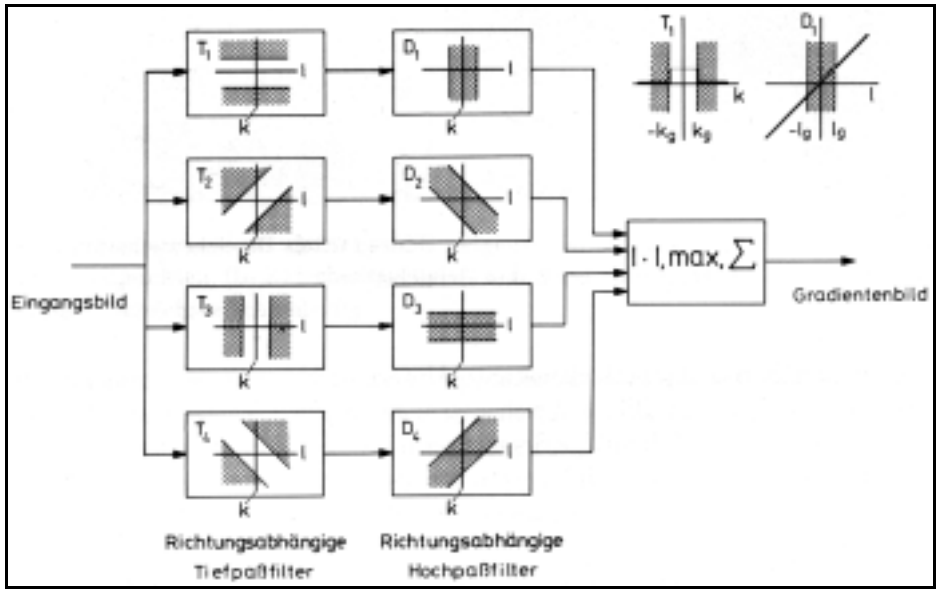


Abbildung 5.7: Prinzip lokaler Gradientenoperatoren.

Die ersten beiden Schritte im Verfahren sind dabei schon erwähnte lineare Operatoren. Zwei Operatoren und ihre zugehörigen Faltungskerne werden im folgenden zum besseren Verständnis erläutert.

Der sehr gebräuchliche Sobel-Operator arbeitet mit einem 3×3 Pixel großen Operatorfenster. Er ist relativ unempfindlich gegen verrauschte Bilder, was jedoch mit etwas verbreiterten Konturen im Gradientenbild erkauft wird. Der Sobel-Operator berechnet Differenzen erster Ordnung und ist definiert durch einen horizontalen und einen vertikalen Faltungskern:

$$\kappa_{hor} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \qquad \kappa_{ver} = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \tag{5.12}$$

Die Beträge der Faltungskerne werden zum Ergebnis addiert: $g_{sobel} = |\kappa_{hor}| + |\kappa_{ver}|$.

(a) **(b)**

Abbildung 5.8: Beispielbild (a), Anwendung des Sobel-Operators (b).

Die Schwächen des Sobel-Operators liegen bei diagonal verlaufenden Kanten. Um auch diese sicher zu detektieren, sind mehr als zwei Faltungkerne notwendig, wie schon im Diagramm der Abbildung angedeutet. Der Kompaßgradient arbeitet sogar mit 8 Faltungskernen, einen für jede Himmelsrichtung. Beispielhaft seien hier die Faltungskerne der Richtungen Süd und Südost angegeben, die anderen entstehen durch „Rotation" der Gewichtungen:

$$\kappa_s = \begin{bmatrix} -1 & -1 & -1 \\ 1 & -2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad \kappa_{se} = \begin{bmatrix} -1 & -1 & 1 \\ -1 & -2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad ... \tag{5.13}$$

Weil alle 8 Richtungsgradienten einzeln berechnet werden müssen und anschließend wieder zu einem Gradientenbild zusammengefügt werden, ist der Kompaßgradient jedoch sehr aufwendig.

5.3 Frequenzraum

Das menschliche Auge nimmt Bilder nicht in gerasterter, digitaler Form wahr, sondern als kontinuierliches Bild. Gleiches gilt i.A. auch für nichtdigitale Aufnahmegeräte, wie z.B. herkömmliche Fotokameras. Stets werden dabei Lichtwellen, die unser Auge (oder ein Objektiv) treffen, in Bildsignale umgerechnet (oder lösen chemische Reaktionen aus). Insbesondere bei medizinischen Aufnahmetechniken sind es häufig nicht Lichtwellen, die ein Bild ausmachen, sondern Wellen anderen Ursprungs, z.B. elektromagnetische Wellen bei der Magnetresonanztomographie, siehe Kapitel 3.1. Zur Darstellung und Bearbeitung von Bildern mit Hilfe von Computern ist daher eine Digitalisierung (mathematisch: Diskretisierung) notwendig. Dabei kommen mathematische Transformationen zum Einsatz, die in der Lage sind, kontinuierliche Wertebereiche auf diskrete Wertebereiche abzubilden.

Sind diese Transformationen reversibel, d.h. wenn $\mathsf{T} : f \to F$ und $\mathsf{T}^{-1} : F \to f$ gilt, so können sie auch dafür benutzt werden, auf Bilder im transformierten Zustand Operationen durchzuführen, die im „nicht-transformierten" Zustand zu aufwendig wären. Beispiele hierfür folgen in den nächsten Abschnitten. In der computergestützten Bildverarbeitung hat sich insbesondere die Fourier-Transformation durchgesetzt, um Bilder zwischen diskreten und kontinuierlichen Raum zu transformieren. In den folgenden Abschnitten wird nun ein kurzer Überblick über die zweidimensionale Fourier-Transformation gegeben, wie sie bei Bilddaten verwendet wird. Hier sollen jedoch zunächst nur Definitionen und einige wichtige, später noch verwendete Eigenschaften angegeben werden.

Das Haupteinsatzgebiet der Fourier-Transformation im Rahmen dieser Arbeit beschränkt sich auf die digitale Filterung. Dabei ist insbesondere die Hin- und Rücktransformation von diskreten Bilddaten

nötig, weswegen bezüglich der Definitionen und Eigenschaften eine kurze mathematische Herleitung der diskreten Fourier-Transformation gegeben wird. Dabei wird auch kurz das Prinzip der Anwendung von Operatoren und Filtern im Frequenzraum erläutert und der Zusammenhang mit dem Ortsraum hergestellt.

5.3.1 Die (kontinuierliche) Fourier-Transformation

Im Rahmen der digitalen Bildverarbeitung wird die Fourier-Transformation sowohl zur Bildgewinnung als auch zur digitalen Filterung eingesetzt. Bei der letzteren versucht man u.a. Grauwertstörungen im Bild zu eliminieren. Diese Grauwertstörungen können z.B. durch systematische Fehler bei der Aufnahme oder der Digitalisierung eines Bildes verursacht worden sein. Ein anderes Anwendungsgebiet der Fourier-Transformation ist die Bildkodierung. Hier wird z.B. untersucht, ob ein digitales Bild weniger Bandbreite bei der Übertragung benötigt, wenn man es anstatt der Grauwerte im Ortsbereich anhand der Fourierkoeffizienten (den Linearfaktoren der Fourier-Transformation) im Frequenzbereich beschreibt. Schließlich wird die Fourier-Transformation auch bei der Charakterisierung von Oberflächenstrukturen oder der Formbeschreibung eines Objektes verwendet.

Sei $f(x,y)$ eine kontinuierliche Funktion der beiden reellen Variablen x,y. Die zweidimensionale, kontinuierliche Fourier-Transformierte $F(u,v)$ von $f(x,y)$ ist dann definiert als

$$F(u,v) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(x,y)e^{-j2\pi(ux+vy)}\,dx\,dy \qquad (5.14)$$

und die obige Gleichung wird als Fourier-Transformation ($\mathsf{F} : f(x,y) \rightarrow F(u,v)$) bezeichnet. Entsprechend dazu läßt sich die Ursprungsfunktion $f(x,y)$ wieder aus der kontinuierlichen, absolut integrierbaren Funktion $F(u,v)$ mittels der zweidimensionalen, kontinuierlichen, inversen Fourier-Transformation berechnen:

$$f(x,y) = \frac{1}{4\pi^2}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} F(u,v)e^{j(ux+vy)}\,du\,dv \qquad (5.15)$$

Die Variablen u und v sowie der Funtkionswert $F(u,v)$ sind dabei im allgemeinen komplexwertig, d.h.es gilt $u,v \in \mathbb{C}$ und $j = \sqrt{-1}$:

$$F(u,v) = \mathrm{Re}\{F(u,v)\} + j \cdot \mathrm{Im}\{F(u,v)\}$$

Die Funktion $f(x,y)$ und ihre Fourier-Transformierte $F(u,v)$ werden dann als Foriertransformationspaar bezeichnet, im folgenden kenntlich gemacht durch das Symbol $\circ\!\!-\!\!\!\stackrel{F}{\bullet}$.

$$f(x,y) \;\circ\!\!-\!\!\!\stackrel{F}{\bullet}\; F(u,v) \qquad (5.16)$$

Dabei deutet der ausgefüllte Kreis in die Richtung der Transformation. Faßt man die beiden Variablen x,y als Ortskoordinaten auf und damit $f(x,y)$ als kontinuierliche Funktion in Ortsraum, so bezeichnet man $F(u,v)$ als kontinuierliches Ortsfrequenzspektrum der beiden Ortsfrequenzvariablen u,v. Das sogenannte Betragsspektrum oder Amplitudenspektrum berechnet sich durch

$$|F(u,v)| = \sqrt{\mathrm{Re}\{F(u,v)\}^2 + \mathrm{Im}\{F(u,v)\}^2} \qquad (5.17)$$

und das Phasenspektrum mit

$$\varphi(u,v) = \arctan\left(\frac{\mathrm{Re}\{F(u,v)\}}{\mathrm{Im}\{F(u,v)\}}\right) \qquad (5.18)$$

Das Quadrat des Betragsspektrums wird als Leistungsspektrum oder Power-Spektrum (engl. power-spectrum) bezeichnet:

$$|F(u,v)|^2 = \mathrm{Re}\{F(u,v)\}^2 + \mathrm{Im}\{F(u,v)\}^2 = F(u,v)\cdot F^*(u,v) \qquad (5.19)$$

Die Notation F^* kennzeichnet hier das konjugiert Komplexe von F.

Zur Veranschaulichung der Fourier-Transformation sind in Abbildung 5.9 einige Beispielbilder und die zugehörigen Fourier-Transformierten angegeben.

Es sei noch erwähnt, daß die Darstellung des Ergebnisses einer Fourier-Transformation meist nicht ohne weiteres möglich ist. Zur qualitativen Beurteilung der in einem Bild vorhandenen Ortsfrequenzen wird daher eine bildliche Darstellung für das (Betrags-) Spektrum $|F(u,v)|$ angestrebt. Die dabei verwendeten Grauwerte werden so gewählt, daß sie zum Betrag der jeweiligen komplexen Spektralkomponente proportional sind. Deswegen benutzt man eine normierte und logarithmierte Darstellung des Betragsspektrums:

$$F_{disp} = c\cdot\log\left(1+|F(u,v)|\right) \qquad (5.20)$$

Daduch werden Spektralkomponenten mit kleinem Betrag etwas stärker betont als solche mit großem Betrag, in Abbildung 5.10b und (c) wird der Unterschied deutlich sichtbar. Durch die Normierung mit einem Faktor c kann sichergestellt werden, daß alle Grauwerte innerhalb der Grauwertmenge des Zielbildes liegen.

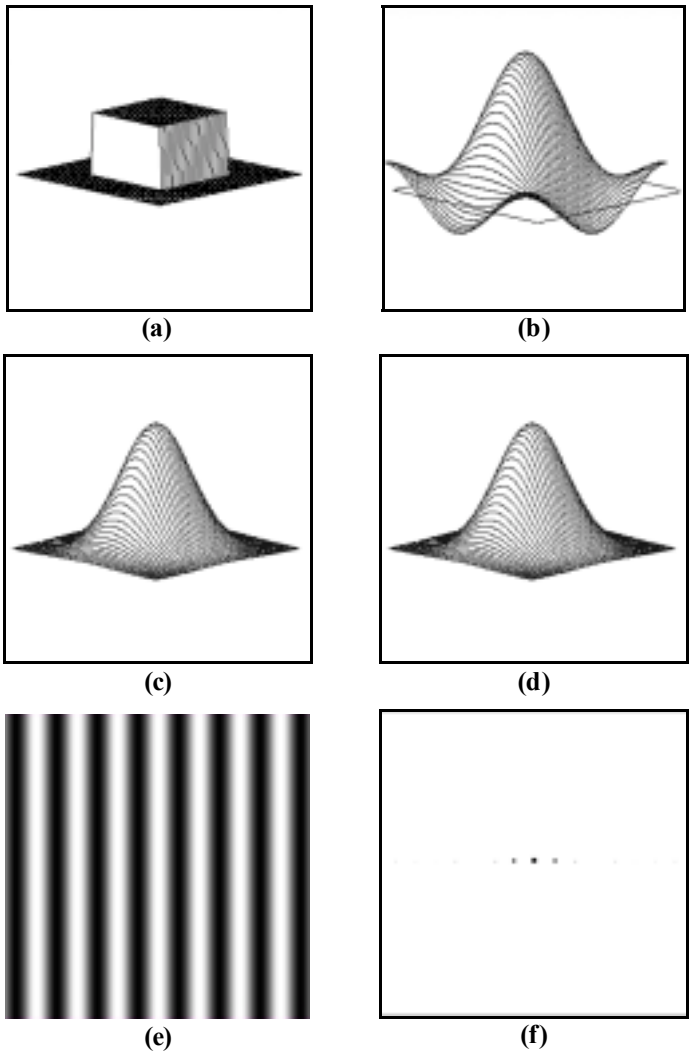
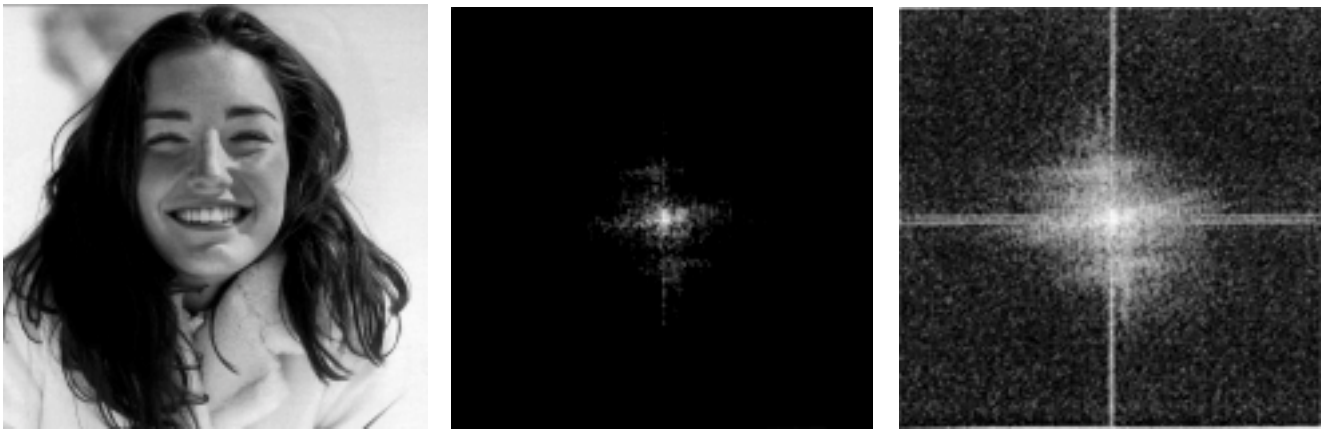


Abbildung 5.9: Beispiele zum Verständnis der Fourier-Transformation [KOPP97]. Rechteckfunktion (a), Fourier-Transformierte der Rechteckfunktion (b), Gaußglocke (c) und deren Transformierte (d), Cosinuswelle (e) und Transformierte (f).



(a) (b) (c)

Abbildung 5.10: Darstellung des Betragsspektrums der Fourier-Transformatierten. Originalbild (a), Betragsspektrum (b) und logarithmiertes Besptragsspektrum (c).

5.3.2 Wichtige Eigenschaften der kontinuierlichen Fourier-Transformation

Vertauschungssatz

Faßt man das Ortsfrequenzspektrum $F(u,v)$ der kontinuierlichen Funktion $f(x,y)$ wieder als eine neue, kontinuierliche Funktion $F(x,y)$ im Ortsraum auf, so hat deren Fourier-Transformierte die an den Koordinatenachsen gespiegelte Form $f(-u, -v)$ des ursprünglichen Signals, multipliziert mit $4\pi^2$. In der Schreibweise mit Fourier-Notation gilt also:

$$f(x,y) \circ\!\!-\!\!\bullet\ F(u,v) \qquad (5.21)$$

$$F(x,y) \circ\!\!-\!\!\bullet\ 4\pi^2 f(-u,-v)$$

Überlagerungssatz

Da die Fourier-Transformation eine lineare Operation ist, ist die Fourier-Transformierte einer Summe von Ortsfunktionen gleich der Summe ihrer Fourier-Transformierten. Es gilt:

$$\sum_i c_i f_i(x,y) \circ\!\!-\!\!\bullet\ \sum_i c_i F_i(u,v) \quad mit\ f_i(x,y) \circ\!\!-\!\!\bullet\ F_i(u,v) \qquad (5.22)$$

$$und\ c_i \in \mathbb{R},\ i = 1,2,3$$

Ähnlichkeitssatz

Eine Drehung der Ortsfunktion $f(x,y)$ in der Ortsebene führt zu einer Skalierung und Stauchung von $F(u,v)$ in der Ortsfrequenzebene und umgekehrt.

$$f(ax,by) \circ\!\!-\!\!\bullet\ \frac{1}{|ab|} F\left(\tfrac{u}{a}, \tfrac{v}{b}\right) \qquad (5.23)$$

$$mit\ a,b \in \mathbb{R}$$

Verschiebungssatz

Eine Verschiebung der Funktion $f(x,y)$ im Ortsraum bewirkt eine lineare Phasendrehung der Funktion $F(u,v)$ im Ortsfrequenzbereich und umgekehrt:

$$f(x-a,y-b) \circ\!\!-\!\!\bullet\ F(u,v) \cdot e^{-j(ua+vb)} \qquad (5.24)$$

$$f(x,y) \cdot e^{j(cx+dy)} \circ\!\!-\!\!\bullet\ F(u-c,v-d)$$

Faltungssatz:

Das Faltungsprodukt zweier zweidimensionaler kontinuierlicher Funktionen korrespondiert mit dem Produkt der Fourier-Transformierten dieser Funktionen und umgekehrt:

$$f(x,y) \otimes h(x,y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(\xi,\eta)\cdot h(x-\xi, y-\eta)\, d\xi\, d\eta \qquad (5.25)$$

$$= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(x-\xi, y-\eta)\cdot h(\xi,\eta)\, d\xi\, d\eta$$

$$\circ\!\!-\!\!\bullet\ F(u,v)H(u,v)$$

$$f(x,y)\cdot h(x,y) \ \circ\!\!-\!\!\bullet\ \frac{1}{4\pi^2} F(u,v) * H(u,v) \qquad (5.26)$$

$$= \frac{1}{4\pi^2}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} F(\xi,\eta)\cdot H(u-\xi, v-\eta)\, d\xi\, d\eta$$

$$= \frac{1}{4\pi^2}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} F(u-\xi, v-\eta)\cdot H(\xi,\eta)\, d\xi\, d\eta$$

Aus dem Faltungssatz resultiert auch das Prinzip der Bildfilterung im Frequenzbereich, siehe Abbildung 5.11. Dabei versucht man die Tatsache auszunutzen, daß numerisch aufwendige Filter im Ortsbereich (weil z.B. mehrere Faltungen nötig sind) durch die einfacheren Multiplikationen im Frequenzbereich zu ersetzen und dadurch Rechenzeit einzusparen. Jedoch muß immer beachtet werden, daß bei dieser Art der Filterung zusätzliche Rechenzeit für die Hin- und Rücktransformation anfällt.

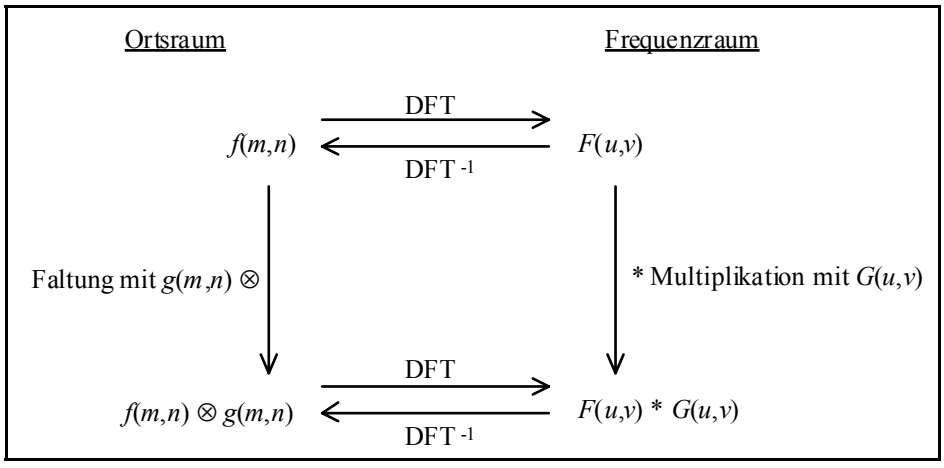


Abbildung 5.11: Prinzip der Filterung im Frequenzbereich.

Korrelationssatz

Auf den Korrelationssatz wird in Abschnitt 6.1 im Zusammenhang mit der Kepstrum-Filterung noch weiter eingegangen.

5.3.3 Die diskrete Fourier-Transformation

Für eine diskrete, zweidimensionale Bildfunktion $f(m,n)$ im Ortsbereich muß zunächst beachtet werden, daß sie nur an ganzzahligen Positionen (entsprechend den Pixeln des durch sie dargestellten Bildes) definiert ist: $m,n \in \mathbb{N}$. Deswegen wird aus dem doppelten Integral der Gleichungen (5.14) und (5.15) für die kontinuierliche Fourier-Transformation eine Doppelsumme. Des weiteren wurde für die kontinuierliche Funktion $f(x,y)$ eine unendliche Ausdehnung angenommen. Diskrete Bilder haben natürlich keine unendliche Ausdehnung, ihre Positionsvariablen sind über einem endlichen $N \times M$ - Intervall definiert und haben außerhalb des Intervalls identisch den Wert 0, siehe Definition (5.3). Man konstruiert nun aus $f(m,n)$ eine Funktion $\tilde{f}(m,n)$ und stellt sich diese periodisch, unendlich fortgesetzt in m- und n-Richtung vor:

$$\tilde{f}(m+\xi M, n+\eta N) = f(m,n) \qquad mit\ \xi,\eta \in \mathbb{Z} \tag{5.27}$$

Unter Annahme dieser Periodizität kann man die zweidimensionale, diskrete Fourier-Transformation definieren als

$$\tilde{f}(m,n) = \frac{1}{MN} \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} \tilde{F}(k,l) \cdot e^{j2\pi\left(\frac{km}{M}+\frac{ln}{N}\right)} \tag{5.28}$$

Mathematisch genauer muß man von der Darstellung von $\tilde{f}(m,n)$ als Fourier-Reihe sprechen. Die Summation ist hier nur im zweidimensionalen Intervall $\{0 \le k \le M-1 \cap 0 \le l \le N-1\}$ durchzuführen, da die Exponentialfunktion in Gleichung (5.28) ebenfalls eine periodische Funktion in den Variablen k,m und l,n ist, siehe auch Gleichung (5.35). Es kann leicht gezeigt werden, daß mit

$$\tilde{F}(k,l) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \tilde{f}(m,n) \cdot e^{-j2\pi\left(\frac{km}{M}+\frac{ln}{N}\right)} \tag{5.29}$$

die Gleichung (5.28) erfüllt ist. Aus Gleichung (5.29) ist unmittelbar ersichtlich, daß $\tilde{F}(k,l)$ die gleiche Periodizität aufweist wie $\tilde{f}(m,n)$, und somit gilt:

$$\tilde{F}(k+\xi M, l+\eta N) = \tilde{F}(k,l) \qquad mit\ \xi,\eta \in \mathbb{Z} \tag{5.30}$$

Die diskrete zweidimensionale Fouriertransfomation wird nun mit Hilfe der Fensterfunktion

$$R_{MN}(\xi,\eta) = \begin{cases} 1 & für\ 0 \le \xi \le M-1 \cap 0 \le \eta \le N-1 \\ 0 & sonst \end{cases} \tag{5.31}$$

definiert als $F(k,l) = \tilde{F}(k,l) \cdot R_{MN}(k,l)$, die inverse, zweidimensionale Fourier-Transformation stellt sich entsprechend dar als $f(m,n) = \tilde{f}(m,n) \cdot R_{MN}(m,n)$. Bei der numerischen Berechnung ist es selbstverständlich ausreichend, sich auf die Berechnung jeweils einer Periode bzw. auf die Berechnung innerhalb des Fensters $R_{MN}(m,n)$ bzw $R_{MN}(k,l)$ zu beschränken. Zur Vereinfachung schreibt man für die diskrete Fourier-Transformation im allgemeinen

$$F(k,l) = \begin{cases} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) \cdot e^{-j2\pi\left(\frac{km}{M}+\frac{ln}{N}\right)} \\ \quad für\ 0 \le k \le M-1 \cap 0 \le l \le N-1 \\ 0 \quad sonst \end{cases} \tag{5.32}$$

bzw. für die inverse diskrete Fourier-Transformation

$$f(m,n) = \begin{cases} \frac{1}{MN} \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} F(k,l) \cdot e^{j2\pi\left(\frac{km}{M}+\frac{ln}{N}\right)} \\ \quad für\ 0 \le m \le M-1 \cap 0 \le n \le N-1 \\ 0 \quad sonst \end{cases} \tag{5.33}$$

oder in Kurzschreibweise

$$f(m,n) \circ\!\!-\!\!\!\stackrel{DF}{\bullet}\!\!- F(k,l) \qquad (5.34)$$

wobei der ausgefüllte Kreis wieder in Richtung der Transformation deutet.

5.3.4 Wichtige Eigenschaften der diskreten Fourier-Transformation

Wird das Abtasttheorem bei der Bilddiskretisierung (Bildgewinnung) erfüllt, so gelten die Eigenschaften der kontinuierlichen Foriertransformation auch für die diskrete Fourier-Transformation [ENCA98], [MÜLL95]. Das Abtasttheorem besagt, daß bei der Abtastung einer kontinuierlichen Bildfunktion die Abtastfrequenz mindestens doppelt so hoch sein muß wie die Bandbreite, also die höchste im Bild auftretende Frequenz [WAHL98]. Nur dann ist eine Rekonstruktion des kontinuierlichen Bildes aus dem diskreten Bild möglich. Gleiches gilt auch für die Erzeugung eines diskreten Bildes aus den *k*-Raum-Daten eines Magentresonanztomographen. Zwar werden hier andere Abtastfunktionen verwendet, jedoch bleibt die Forderung an die Abtastfrequenz erhalten [HASH97].

Die Gleichungen sind für die diskrete Fouriertransformation entsprechend sehr ähnlich, weswegen auf eine explizite Angabe an dieser Stelle verzichtet wird. Sollten Gleichungen für Eigenschaften der diskreten Fourier-Transformation in den folgenden Kapiteln benötigt werden, so werden sie dort angegeben. Eine ausführliche Herleitung der Eigenschaften im Diskreten kann in [WAHL89] gefunden werden.

5.3.5 Herleitung der diskreten Fourier-Transformation

Aus der Mathematik ist die Möglichkeit bekannt, Funktionen als endliche oder unendliche Reihen darzustellen. Ein bekanntes Beispiel ist die Reihenentwicklung für e^x:

$$e^x = 1 + \frac{1}{1!}x + \frac{1}{2!}x^2 + \frac{1}{3!}x^3 + \ldots \qquad (5.35)$$

Die Funktionen x^n werden als Basisfunktionen bezeichnet, die in der Reihenentwicklung mit bestimmten Linearfaktoren auftreten. Von besonderem Interesse sind solche Systeme von Basisfunktionen, die orthonomiert sind; eine Eigenschaft, die im folgenden noch erklärt wird. Legt man nun eine Menge von Basisfunktionen fest, so kann man eine Funktion $f(x)$ auch durch die Linearfaktoren beschreiben, die in der Reihenentwicklung auftreten. Es zeigt sich, daß so manche Eigenschaften der Funktion $f(x)$ anhand der Linearfaktoren der transformierten Funktion $F(x)$ besser zu erkennen sind als in der Originalfunktion. Für kontinuierliche und diskrete Funktionen ist die Fourier-Transformation ein wichtiges Beispiel einer derartigen Anwendung.

Sei nun mit $\mathbb{C}_{MN}$ die Menge aller $M{\cdot}N$-Matrizen bezeichnet, wobei die Elemente einer Matrix auch komplexwertige Zahlen sein dürfen. $\mathbb{C}_{MN}$ wird genau dann zu einem linearen Raum über dem Körper $\mathbb{C}$ der komplexen Zahlen, wenn die Addition $\oplus$ zweier Matrizen S und T und die linearen Operationen mit Elementen a, b aus $\mathbb{C}$ definiert werden. Es gilt dann:

$$S \oplus T = [s(m,n)] + [t(m,n)] = [s(m,n) + t(m,n)], \qquad (5.36)$$

$$(ab)S = a(bS) = a[b \cdot s(m,n)] = [ab \cdot s(m,n)],$$

$$\text{sowie } (a+b)S = aS + bS \text{ und } a(S+T) = aS + aT$$

Schließlich wird noch das Skalarprodukt $\otimes$ definiert:

$$S \otimes T = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} s(m,n) \cdot t^*(m,n) \qquad (5.37)$$

Wobei mit t^* wieder die konjugiert komplexe Zahl zu $t(m,n)$ bezeichnet wird. Damit ist $\mathbb{C}_{MN}$ ein unitärer Raum über dem Körper $\mathbb{C}$ der komplexen Zahlen. Die Elemente S von $\mathbb{C}_{MN}$ sind als Linear-kombination von $K=M\cdot N$ linear unabhängigen Basismatrizen (einer Basis) darstellbar:

$$B^{(u,v)} = [b(m,n)]^{(u,v)} \text{ mit } u=0,...,M\text{-}1 \text{ und } v=0,...,N\text{-}1 \qquad (5.38)$$

$$s(m,n) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} f(u,v) \cdot b(m,n)^{(u,v)} \text{ mit } m=0,...,M\text{-}1 \text{ und } n=0,...,N\text{-}1$$

Für eine orthonomierte Basis muß gelten:

$$B^{(u_1,v_1)} \otimes B^{(u_2,v_2)} = 0, \quad \text{falls } (u_1,v_1) \neq (u_2,v_2) \quad \text{und} \qquad (5.39)$$
$$B^{(u_1,v_1)} \otimes B^{(u_2,v_2)} = 1, \quad \text{falls } (u_1,v_1) = (u_2,v_2)$$

Es läßt sich zeigen, daß die Basismatrizen

$$B^{(u,v)} = \left[e^{i 2\pi \left(\frac{mu}{M} + \frac{nv}{N} \right)} \right] \text{ mit } u=0,...,M\text{-}1 \text{ und } v=0,...,N\text{-}1 \qquad (5.40)$$

linear unabhängig und orthonomiert sind. Damit kann eine $M\cdot N$-Matrix $S = [s(m,n)]$ als Linearkombination dieser Basismatrizen geschrieben werden:

$$s(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} f(u,v) \cdot e^{j 2\pi \left(\frac{ux}{M} + \frac{vy}{N} \right)} \qquad (5.41)$$

$$\text{für } x = 0,...,M\text{-}1 \text{ und } y = 0,...,N\text{-}1$$

Da die Basismatrizen orthonomiert sind, berechnen sich die Linearfaktoren $f(u,v)$ gemäß:

$$f(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{x=0}^{N-1} s(x,y) \cdot e^{-j 2\pi \left(\frac{ux}{M} + \frac{vy}{N} \right)} \qquad (5.42)$$

$$\text{für } u = 0,...,M\text{-}1 \text{ und } v = 0,...,N\text{-}1$$

Die in (5.40) definierten Matrizen sind die Basis der diskreten, zweidimensionalen Fourier-Transformation, (5.42) entspricht nun (5.32) und wird als diskrete, zweidimensionale Fourier-Transformation bezeichnet. Die Gleichung (5.41) wird als inverse, diskrete, zweidimensionale Fourier-Transformation bezeichnet und entspricht damit (5.33).

5.4 Interpolation

Geometrische Transformationen sind auf einem digitalen Bildraster nur in Sonderfällen direkt durchführbar. Beispiele solcher Sonderfälle sind pixelweise Verschiebungen oder Rotationen um $n\cdot 90°$, $n \in \mathbb{N}$. Im allgemeinen passen die transformierten Pixelkoordinaten jedoch nicht mehr auf die ursprünglichen Rasterkoordinaten. Dann muß ein neues Pixel aus den Werten seiner Nachbarn berechnet werden; dieser Vorgang wird als Interpolation bezeichnet. Die Qualität ist dabei sowohl für die Genauigkeit der Kepstrum-Filterung (siehe Kapitel 6.2) als auch für die anschließende Extraktion quantitativer Parameter für die medizinsche Diagnostik von besonderer Bedeutung. In dieser Arbeit wird das gleiche Interpolationsverfahren sowohl zur Verschiebung von Bildern um reelle Pixelkoordinaten als auch zur Skalierung genutzt.

Interpolation durch Pixelwiederholung kann nur für ganzzahlige Skalierungswerte angewendet werden und wird häufig zur schnellen Darstellung von Vergrößerungen benutzt (Lupenfunktion, etc). Auch die bilineare Interpolation, bei der Grauwerte durch einfache Mittelung aus den Nachbarpixeln berechnet werden, genügt den geforderten Qualitätsansprüchen an die Qualität nicht. Im folgenden wird daher eine Einführung in die auf Spline-Kurven basierende kubische Interpolation gegeben [ABRA91].

5.4.1 Interpretation als Faltung

Die Interpolation kann in zwei Schritten interpretiert werden. Zuerst wird aus dem vorhandenen diskreten Bild $f(m,n)$, $m,n \in \mathbb{N}$ eine kontinuierliche Bildfunktion $f(x,y)$, $x,y \in \mathbb{R}$ erzeugt und diese anschließend auf dem neuen Raster wieder abgetastet. Das Ergebnis ist wieder ein diskretes Bild.

Die Berechnung der kontinuierlichen Koordinatenwerte aus den diskreten Nachbarn kann formal als Faltung mit einem kontinuierlichen, zweidimensionalen Faltungskern $h(x,y)$ aufgefaßt werden, vgl. auch (5.25):

$$f(x,y) = \sum_m \sum_n f(m,n) \cdot h(x-m, y-n) \qquad (5.43)$$

Dabei werden symmetrische und separierbare Interpolationsfunktionen verwendet, um den Rechenaufwand zu minimieren:

$$h(x,y) = h(x) \cdot h(y) \qquad (5.44)$$

Das Vorgehen bei einer separiert durchgeführten Interpolation in einer 4×4-Nachbarschaft (K=4 Stützstellen) wird in Abbildung 5.12 gezeigt. Zunächst werden mit dem eindimensionalen Kernel $h(y)$ zeilenweise Stützwerte berechnet, die in der Abbildung klein und grau dargestellt sind. Die dabei durchgeführte gewichtete Addition der schwarz dargestellten Pixel entspricht der Faltung nach (5.43) im Eindimensionalen. Aus den Stützwerten ergibt sich dann durch eine Faltung mit demselben Kernel in x-Richtung der gesuchte Interpolationswert, in der Abbildung groß und grau dargestellt.

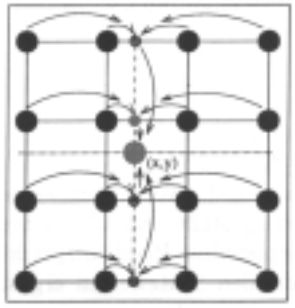


Abbildung 5.12: Berechnung der kubischen Interpolation [LEHM98].

An die Interpolationsfunktion können a-priori einige Anforderungen gestellt werden. Fallen die zu berechnenden Interpolationswerte genau auf die vorhandenen diskreten Pixelwerte, soll durch die Interpolation der Pixelwert nicht verändert werden. Damit muß für jede Funktion $h(x)$ gelten:

$$h(0) \equiv 1, \quad und \quad h(x) \equiv 0, \quad \forall x \in \mathbb{Z} \setminus \{0\} \qquad (5.45)$$

Weiterhin sollte die Grundhelligkeit des Bildes durch die Interpolation nicht verändert werden. Daher muß die Summe der Abtastungen der Interpolationsfunktion in ganzzahligen Abständen für beliebige Verschiebungen d im Intervall $0 \le d < 1$ stets 1 sein:

$$\sum_{k=0}^{K-1} h\left(d + \left(k - \frac{K}{2}\right)\right) \equiv 1, \qquad K = 1,2,4,6,8,... \qquad (5.46)$$

In Gleichung (5.46) bezeichnet K die Anzahl der Stützstellen in einer Dimension der K×K-Nachbarschaft. Für K=1 ergibt sich die sog. Nearest-Neighbour-Interpolation: Es wird einfach der Grauwert desjenigen Pixels genommen, dessen Abstand zum neuen Rasterpunkt am kleinsten ist.

5.4.2 Die kubische Interpolation

Bei der kubischen Interpolation werden Polynome dritter Ordnung jeweils an den ganzzahligen Integerwerten $\lfloor x \rfloor \in \mathbb{N}$ stückweise aneinander gesetzt [ABRA91]. Eine kubische Interpolation mit

einer Stützstelle (K=1) ist nicht definiert. Für die symmetrische kubische Interpolation mit K=2 Stützstellen $h_2(x)$ folgt mit $A_i \in \mathbb{R}$:

$$h_2(x) = \begin{cases} A_3|x|^3 + A_2|x|^2 + A_1|x| + A_0 & 0 \le |x| \le 1 \\ 0 & sonst \end{cases} \qquad (5.47)$$

Die Parameter A_0 bis A_3 können aus den Randbedingungen (5.45) sowie der Forderung nach stetig differrenzierbaren Übergängen an den Stellen $\lfloor x \rfloor$ bestimmt werden. Mit der Notation $h(x^-)$ und $h(x^+)$ für den links- bzw rechtsseitigen Grenzwert der Funktion h an der Stelle x ergibt sich für K=2:

$$\begin{aligned} h_2(0) = 1 \quad &\Rightarrow \quad A_0=1 \\ h_2(1^-) = h_2(1^+) \quad &\Rightarrow \quad A_3+A_2+A_1+A_0=0 \\ h'_2(0^-) = h'_2(0^+) \quad &\Rightarrow \quad -A_1=A_1 \\ h'_2(1^-) = h'_2(1^+) \quad &\Rightarrow \quad 3A_3+2A_2+A_1=0 \end{aligned} \qquad (5.48)$$

Und damit folgt dann weiter:

$$h_2(x) = \begin{cases} 2|x|^3 - 3|x|^2 + 1|x| + A_0 & 0 \le |x| \le 1 \\ 0 & sonst \end{cases} \qquad (5.49)$$

Da (5.45) in die Bestimmung der Koeffizienten eingegangen ist, ist das Kriterium (5.45) von Beginn an erfüllt. Weiter kann man leicht zeigen, daß auch Gleichung (5.46) für beliebige $0 \le d < 1$ erfüllt ist.

Der in der Implementierung angewendete Algorithmus benutzt vier Stützstellen (K = 4) und ist numerisch genauer als die Interpolation mit nur 2 Stützstellen. Werden 4 Stützstellen genutzt, können aus den Randbedingungen sieben der acht Parameter bestimmt werden, so daß zunächst ein freier Parameter α unbestimmt bleibt:

$$h_4(x) = \begin{cases} (a+2)|x|^3 - (a+3)|x|^2 + 1 & 0 \le |x| < 1 \\ a|x|^3 - 5a|x|^2 + 8a|x| - 4a & 1 \le |x| \le 2 \\ 0 & sonst \end{cases} \qquad (5.50)$$

Unabhängig von a ist Gleichung (5.46) immer erfüllt, denn mit K = 4 ergibt sich aus (5.46) und Gleichung (5.50):

$$\begin{aligned} \sum_{k=0}^{3} h(d+k-2) &= h_4(d-2) + h_4(d-1) + h_4(d) + h_4(d+1) \qquad (5.51) \\ &= a(8-12d+6d^2-d^3) - 5a(4-4d+d^2) + 8a(2-d) - 4a \\ &\quad + (a+2)(1-3d+3d^2-d^3) - (a+3)(1-2d+d^2) + 1 \\ &\quad + (a+2)d^3 - (a+3)d^2 + 1 \\ &\quad + a(d^3+3d^2+3d+1) - 5a(d^2+2d+1) + 8a(d+1) - 4a \\ &\equiv 1 \end{aligned}$$

Die Wahl des freien Parameters a ist durchaus nicht unkritisch für die Qualität der Interpolation. Untersuchungen haben gezeigt, daß a entscheidenden Einfluß auf den Fehler bei der Interpolation hat. Daher wurden von verschiedenen Autoren [KEYS81], [PARK83], [HER94] verschiedene Konzepte zur Bestimmung vorgeschlagen, im Rahmen dieser Arbeit $a = -\frac{1}{4}$ gewählt wird.

Auf die Interpolation mit 6 oder 8 Stützstellen soll hier nicht weiter eingegangen werden. In der Praxis sind 4 Stützstellen völlig ausreichend, da der Gewinn an Genauigkeit im Vergleich zur kubischen Interpolation mit mehr Stützstellen zumindest in diesem Anwendungsfall in keinem Verhältnis zum erhöhten Rechenaufwand steht. Weitere Informationen sowie ein qualitativer Vergleich verschiedener Verfahren hinsichtlich des numerischen Aufwandes und Interpolationsfehlers sind in [LEHM98] zu finden.

6 Kepstrum-Filterung zur Translationskorrektur

Wie schon in Kapitel 4 ausgeführt wurde, bezeichnet die Informatik das Problem des Vergleichs zweier Bilder mit gleichem oder ähnlichen Inhalt allgemein mit dem Begriff Registrierung. In den meisten Fällen, soll dabei ein Bild anhand eine Referenzbildes justiert, d.h. geometrisch ausgerichtet werden. Dieses Problem tritt insbesondere bei medizinischen Bilddaten häufig auf. Bei der vergleichenden Befundung wird es durch die Vorstellungsgabe des Arztes gelöst, jedoch erfordern Bildverarbeitungssysteme zur automatischen Befundunterstützung eine algorithmische Lösung.

Beispiele hierfür findet man nicht nur in der Subtraktionsangiographie oder Videoendoskopie, wo die zu vergleichenden Aufnahmen in kurzen Abständen hintereinander erzeugt werden, sondern auch bei der radiographischen Verlaufskontrolle von Implantationen, bei denen Zeiträume von 6-12 Monaten zwischen den einzelnen Wiederholungs-Aufnahmen (engl. recall images) liegen können [CIDE95], [GERL89].

Im einfachsten Fall beschränken sich die Unterschiede in der Aufnahmegeometrie auf Verschiebungen entlang der x- und y-Achse. Der für diesen Fall klassische Algorithmus basiert auf der Kreuz-Korrelationsfunktion und wird in der Bildverarbeitung oft auch *template matching* genannt. Die Kreuz-Korrelation sowie Methoden zur Messung der Ähnlichkeit zweier Bilder werden zu Beginn dieses Kapitels kurz behandelt.

Nur numerisch effiziente Algorithmen können auch auf Bildsequenzen mit sehr vielen Einzelbildern angewendet werden. Da die Kreuz-Korrelation äußerst rechenaufwendig ist, eignet sie sich nicht sehr gut für die Nierensequenzen dieser Arbeit. Der zweite, größere Abschnitt dieses Kapitels widmet sich daher mit der Kepstrum-Technik einem effizienteren Verfahren. Es basiert auf der Fourier-Transformation, arbeitet im Frequenzraum (radiologisch k-Raum) und bietet die schon in der Einleitung zu Kapitel 4.3.1 erwähnten Vorteile der Unempfindlichkeit gegenüber Rauschen und Unabhängigkeit von dargestellten Objekten.

Um die Kepstrum-Filterung optimal an die Nierensequenzen anzupassen, sind noch einige Erweiterungen notwendig, deren Methoden im Anschluß an die Darstellung der Kepstrum-Technik erläutert werden.

6.1 Korrektur von Translationen

Seien $f_1(x,y)$ und $f_2(x,y)$, $x,y \in \mathbb{R}$, zwei endliche kontinuierliche Bildfunktionen, die eine Aufnahme des gleichen Objektes darstellen. Diese sind nun um die Distanz (δ_x,δ_y) gegeneinander verschoben:

$$f_2(x,y) = f_1(x-\delta_x, y-\delta_y) \qquad (6.1)$$

Zur Bestimmung der Verschiebung wird eine Funktion $c(x,y)$ benötigt, die an der Stelle (δ_x,δ_y) ein mathematisches Extremum, z.B. das globale Maximum besitzt:

$$c(x,y) < c(\delta_x,\delta_y), \qquad \forall x \neq \delta_x, y \neq \delta_y \qquad (6.2)$$

Damit ist die Verschiebung dann einfach durch (partielle) Ableitung bestimmbar.

6.1.1 (Kreuz-)Korrelation

Die Korrelation zweier endlicher kontinuierlicher Funktionen $f_1(x)$ und $f_2(x)$, $x \in \mathbb{R}$, ist definiert durch die Gleichung

$$c(x) = \int_{-\infty}^{+\infty} f_1^*(a) f_2(x+a)\, da \qquad (6.3)$$

wobei f^* die konjugiert komplexe Funktion von f ist. Anschaulich kann man sich vorstellen, daß $f_1(x)$ über den gesamten Bereich von $f_2(x)$ „geschoben" wird und das Integral von $+\infty$ bis $-\infty$ des Produkts beider Funktionen für jede Verschiebung a berechnet wird. Sind die Funktionen $f_1(x)$ und $f_2(x)$ gleich, bezeichnet man die Gleichung (6.3) als Autokorrelationsfunktion, sind $f_1(x)$ und $f_2(x)$ unterschiedlich, so wird sie Kreuzkorrelationsfunktion genannt.

Allgemein gibt es viele Möglichkeiten, die Korrelationsfunktion herzuleiten. Eine anschauliche Methode ist der signaltheoretische Ansatz. Dabei werden Bilder $f(x,y)$ als zweidimensionale Signale mit endlichem Betrag B aufgefaßt:

$$B = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f^2(x,y)\; dx\, dy < \infty \qquad (6.4)$$

Errechnet man nun die Differenz zweier Bilder $f_1(x,y)$ und $f_2(x,y)$, so kann man deren Betrag B_{diff} als Maß für die Abweichung der beiden Funktionen benutzen:

$$B_{diff} = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} (f_1(x,y) - f_2(x,y))^2\; dx\, dy < \infty \qquad (6.5)$$

Mit Hilfe der binomischen Formeln ergibt sich aus (6.5)

$$B_{diff} = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f_1^2(x,y)\, dx\, dy + \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f_2^2(x,y)\, dx\, dy - 2\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f_1(x,y)\, f_2(x,y)\, dx\, dy \qquad (6.6)$$

wobei die ersten beiden Terme in Gleichung (6.6) den Einzelbeträgen der Bildfunktionen f_1 und f_2 entsprechen und konstant sind. Der dritte Term in (6.6) wird als Korrelationskoeffizient bezeichnet. Normiert man den Korrelationskoeffizienten auf die Beträge der Bildfunktionen f_1 und f_2, so nimmt er genau dann einen maximalen Wert an, wenn beide Funktionen den gleichen Verlauf haben und sich höchstens um eine reelle Konstante unterscheiden.

Die Abhängigkeit des Korrelationskoeffizienten von der Bildverschiebung führt wieder zur Korrelationsfunktion (6.3), diesmal im Zweidimensionalen:

$$c(x,y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f_1^*(a,b) f_2(x+a, y+b)\; da\, db \qquad (6.7)$$

Einsetzen von Gleichung (6.1) in (6.7) liefert dann

$$c(x,y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f_1^*(a,b) f_1(x+a-\delta_x,\, y+b-\delta_y)\; da\, db \qquad (6.8)$$

Somit kann durch Bestimmung des maximalen Wertes des Korrelationskoeffizienten die Verschiebung bestimmt werden:

$$c(x,y)\big|_{\max} = c(\delta_x,\delta_y) \qquad (6.9)$$

Mit der (Kreuz-)Korrelationsfunktion kann man somit unter Berücksichtigung des Kriteriums (6.2) die relative Verschiebung (δ_x,δ_y) zweier Bildfunktionen gegeneinander berechnen.

6.1.2 Diskrete Ähnlichkeitsmaße

Die im vorigen Abschnitt diskutierte Korrelationsfunktion für den kontinuierlichen Bereich kann natürlich auch im Diskreten angewendet werden. Für die (endlichen) diskreten Bildfunktionen $f_1(m,n)$ und $f_2(m,n)$ mit $0 \le m < M$ und $0 \le n < N$ ergibt sich aus Gleichung (6.7) :

$$c(k,l) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f_1(m,n) f_2(m+k,n+l) \qquad (6.10)$$

Dabei muß für jede Verschiebung (k,l) der Bilder gegeneinander sichergestellt sein, daß nur solche Pixelkoordinaten bei der Berechnung berücksichtigt werden, die im Definitionsbereich der Bilder f_1 und f_2 liegen.

Zur Normierung des diskreten Korrelationskoeffizienten auf die Beträge der Bilder subtrahiert man von den Bildern ihren Mittelwert g_{av}, berechnet nach Gleichung (5.4). Dies führt auf den diskreten (Kreuz-)Kovarianzkoeffizienten c_{CCC}, auch häufig als normierter Korrelationskoeffizient (engl. cross correlation coefficient) bezeichnet:

$$c_{CCC} = \left| \frac{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}(f_1(m,n)-g_{av\,1})(f_2(m,n)-g_{av\,2})}{\sqrt{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}(f_1(m,n)-g_{av\,1})^2 \;\; \sum_{m=0}^{M-1}\sum_{n=0}^{N-1}(f_2(m,n)-g_{av\,2})^2}} \right| \qquad (6.11)$$

Die Korrelationsfunktion ist numerisch äußerst aufwendig, da die Berechnung von (6.11) für jede mögliche Verschiebung der Bilder neu durchgeführt werden muß. Für die Translationsberechnung der dieser Untersuchung zugrundeliegenden Aufnahmeserien wäre sie zwar durchaus geeignet, aber zu langsam. In der Literatur werden deswegen zahlreiche Varianten der Korrelationsfunktion vorgeschlagen, die numerisch weniger aufwendig sind.

Ein Großteil alternativer Verfahren zur Bestimmung der Ähnlichkeit zweier Bilder basiert auf dem Differenzbild $f_{diff}(m,n) = f_1(m,n) - f_2(m,n)$, vergleiche Kapitel 5.1.3. Genau wie der Korrelationskoeffizient muß auch das Differenzbild für jede mögliche Relativposition der Bilder zueinander neu berechnet werden. Abhängig vom Überlappungsbereich, der kleiner oder gleich der Größe der Bilder f_1 und f_2 sein kann, sind dabei bis zu $M \cdot N$ Operationen pro Berechnung eines Differenzbildes notwendig. Anschließend wird z.B. der mittlere Grauwert (5.4) des Differenzbildes berechnet; die Verschiebung des Differenzbildes mit dem minimalen mittleren Grauwert entspricht der gesuchten Translation.

Eine weitere Reduzierung der Rechenoperationen kann durch die Betrachtung des Histogramms des Differenzbildes erreicht werden. Wie bereits erwähnt, kann das Histogramm auch als Aussagekriterium über den Bildinhalt herangezogen werden, siehe Kapitel 5.1.1. Wenn beide Bilder pixelweise identisch sind, d.h. wenn gilt $f_1(m,n) = f_2(m,n)\;\; \forall m,n$, ist das Differenzbild an jeder Position Null und die Histogrammfunktion liefert an der Stelle 0 einen (von Null verschiedenen) hohen Wert und sonst 0. Je mehr sich die Bilder unterscheiden, desto mehr enthält das Differenzbild Werte ungleich 0, und das Histogramm (des Differenzbildes) wird breiter und flacher.

Ein weiteres Verfahren, das den Bildinhalt des Differenzbildes berücksichtigt, wurde 1994 von SAMARABANDU et al. entwickelt und soll hier kurz vorgestellt werden. Ziel war dabei die

automatischen Registrierung dentaler Radiographien. Insbesondere bei kleinen Verschiebungen der Bilder gegeneinander entstehen Kanten im Differenzbild. Dies wird mit (zwei) richtungsselektiven Kantendetektoren gefaltet und die Ergebnisbilder addiert. Da die speziellen Faltungskerne als Koeffizienten nur Werte aus $\{-1,0,+1\}$ enthalten, kann das Verfahren ohne Verwendung von Multiplikationen implementiert werden, jedoch muß die Faltung wiederum für jede mögliche Verschiebung der Bilder durchgeführt werden [SAMA94].

6.1.3 Korrelation im Frequenzraum

Einleitende Überlegungen zu der Korrelation wurden in Kapitel 6.1.1 angestellt. Die kontinuierliche Darstellung der Korrelation $c(x,y)$ der Funktionen $f_1(x,y)$ und $f_2(x,y)$ läßt sich (mit der symbolischen Verwendung des Operators $\circ_c$ für die Korrelation) beschreiben als:

$$f_1(x,y) \circ_c f_2(x,y) = c(x,y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f_1^*(a,b) f_2(x+a, y+b)\ da\, db$$

Formal hat das Korrelationsdoppelintegral starke Ähnlichkeit mit dem zweidimensionalen Integral für die Faltung eines Bildes mit einem Faltungskern im Frequenzraum, dargestellt durch den Operator $*$:

$$f_1(x,y) * f_2(x,y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f_1^*(a,b) f_2(x-a, y-b)\ da\, db$$

Der Zusammenhang zwischen Faltung und Korrelation kann nun leicht hergestellt werden, in dem man die Integrationsvariablen einfach substituiert durch $a' = -a$ und $b' = -b$:

$$f_1(x,y) \circ_c f_2(x,y) = f_1(-x,-y) * f_2(x,y) \qquad (6.12)$$

Zur Erinnerung sei hier noch einmal die Forurier-Transformierte $F(u,v)$ der Funktion $f(x,y)$ angegeben, siehe auch Kapitel 5.3.1:

$$F(u,v) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(x,y) e^{-j2\pi(ux+vy)} dx\, dy$$

$$f(x,y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} F(u,v) e^{j2\pi(ux+vy)} du\, dv$$

Der Zusammenhang zwischen Korrelation und Faltung (6.12) läßt sich nun mit Hilfe des Spiegelungstheorems für reelle Ortsfunktionen und dem Faltungstheorem der Fourier-Transformation zeigen. Die Korrelationsfunktion $c(u,v)$ stellt sich dann als Multiplikation im Frequenzraum dar:

$$f_1(x,y) \circ_c f_2(x,y) \ \circ\!\!-\!\!\bullet\ F_1^*(u,v) \cdot F_2(u,v) \qquad (6.13)$$

Dabei wird durch das Symbol $\circ\!\!-\!\!\bullet$ wieder die Fourier-Transformation in Richtung des ausgefüllten Kreises dargestellt (siehe Kapitel 5.3.1) und F^* ist wieder das konjugiert Komplexe von F. Die Rücktransformation c_{MF} der rechten Seite in (6.13) entspricht der Korrelationsfunktion nach (6.7):

$$F_1^*(u,v) \cdot F_2(u,v) = C(u,v) \ \bullet\!\!-\!\!\circ\ c_{MF}(x,y) \qquad (6.14)$$

Sie hat ihr Maximum wieder an der Stelle (δ_x, δ_y), welche die Verschiebung zwischen den Bildern f_1 und f_2 bestimmt. In der Literatur wird der Korrelationsfilter im Frequenzraum auch als angepaßter Filter (engl. matched filter) bezeichnet [HORN84].

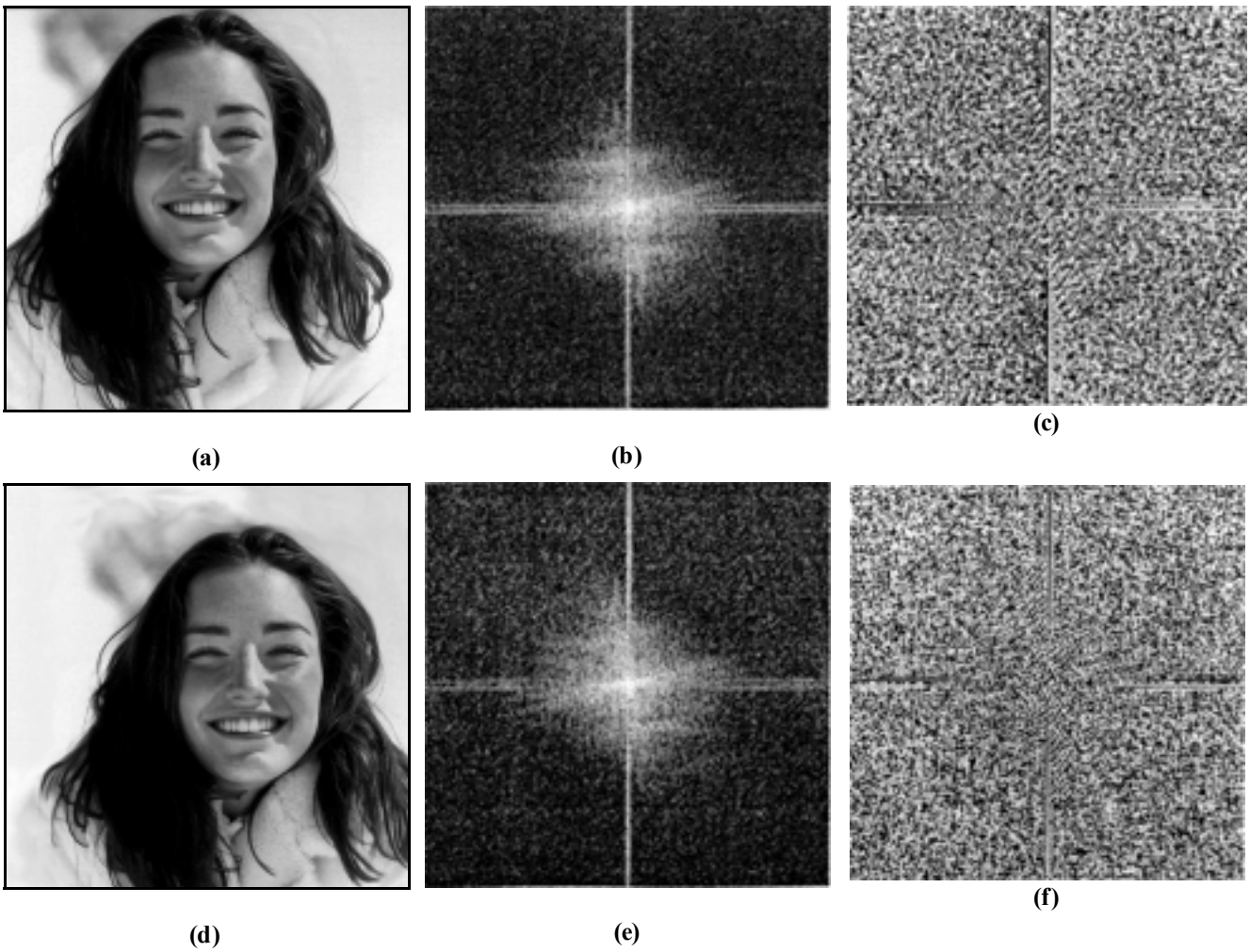


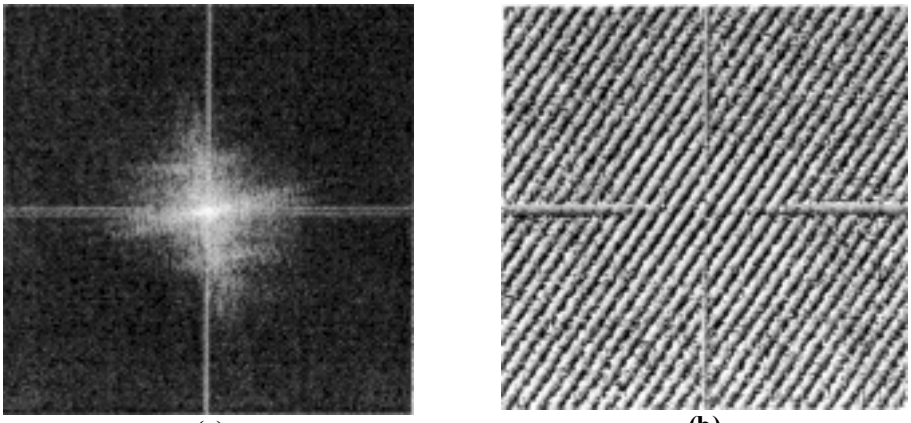
Abbildung 6.1: Beispiel zur Korrelation im Frequenzraum. Bild (d) wurde gegenüber Bild (a) um (18,12) Pixel verschoben. Betrag (e) und Phase (f) des verschobenen Bildes unterscheiden sich kaum vom Referenzbild.

Da der Ablauf der Korrelation im Frequenzraum intuitiv nicht einfach zu verstehen ist, soll das Verfahren im Folgenden an einem Beispiel verdeutlicht werden. Abbildung 6.1 zeigt zwei Ausschnitte $f_1(x,y)$ und $f_2(x,y) = f_1(x-\delta_x,y-\delta_y)$ aus derselben Aufnahme. Die Verschiebung von f_2 zu f_1 beträgt (δ_x,δ_y) = (18,12). Die Fourier-Transformation der beiden Bilder wird nun nach Betrag und Phase zerlegt. Abbildung 6.3a zeigt das Ergebnis des matched filter c_{MF}. Sein Maximum an der Stelle (18,12) ist recht flach, wobei die Form des Maximums von den Amplitudenspektren der Fourier-Transformierten der Funktionen f_1 und f_2 bestimmt wird.

Setzt man nun die Voraussetzung (6.1) vom Beginn dieses Kapitels in (6.13) ein, so ergibt sich mit dem Verschiebungstheorem (5.24) der Fourier-Transformation die folgende Gleichung:

$$F_1^*(u,v)\cdot F_2(u,v) = F_1^*(u,v)\cdot F_1(u,v)\cdot e^{j2\pi(u\delta_x+v\delta_y)} = \left|F_1(u,v)\right|^2\cdot e^{j2\pi(u\delta_x+v\delta_y)} \qquad (6.15)$$

Daraus ist nun direkt ersichtlich, daß die eigentliche Verschiebungsinformation ausschließlich in der Phase des Korrelationssignales kodiert ist, siehe auch Abbildung 6.2b. Ganz allgemein wird die Bedeutung der Phase der Fourier-Transformierten häufig unterschätzt und nur die Amplitude dargestellt. Die Phase enthält aber die Informationen über die Positionen der Grauwertschwankungen (Frequenzen) im Bild, so daß eine annähernde Rekonstruktion des Bildes allein aus der Phase möglich ist. Im Gegensatz dazu ist die Rekonstruktion eines Bildes allein aus den Amplitudenwerten nicht möglich (siehe Abbildung 5.10).



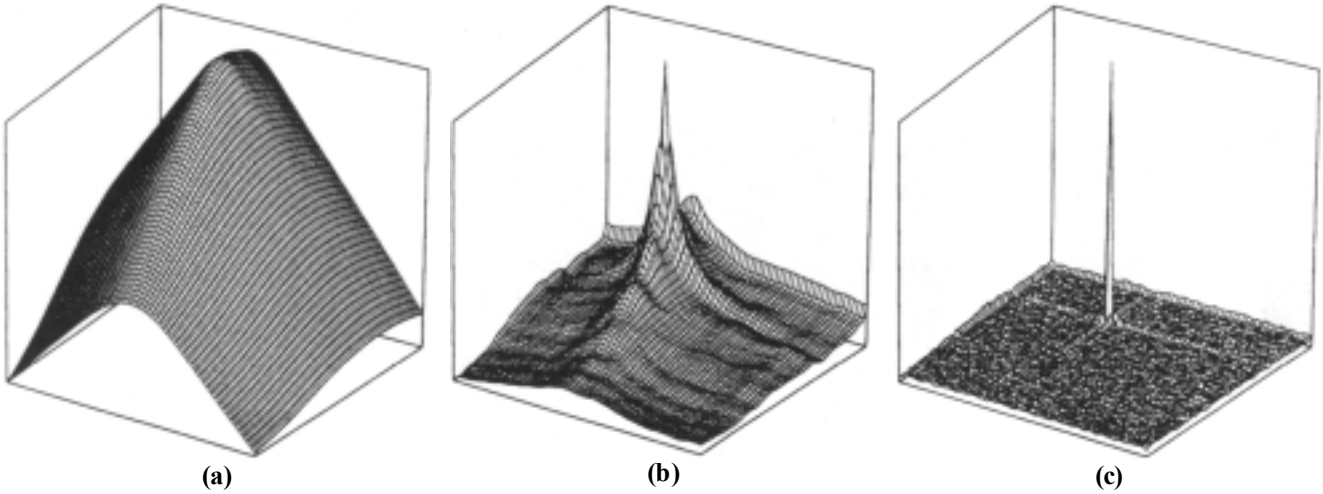
(a) (b)

Abbildung 6.2: Betrag (a) und Phase (b) des Korrelationssignales $F_1^* \cdot F_2$. Der Betrag hat sich im Vergleich zu den Spektren in Abbildung 6.1 kaum geändert. Die Phase weist jedoch eine deutliche Periodizität auf, in der die Verschiebung kodiert ist.

Diese Eigenschaften motivieren dazu, zur Korrelation lediglich die Phase zu benutzen. So werden bei der Phasenkorrelation (engl. phase only matched filter) vor der Rücktransformation der Betrag des Korrelationssignals $F_1^* \cdot F_2$ auf den Betrag der Fourier-Transformierten des Referenzbildes normiert:

$$\frac{F_1^*(u,v) \cdot F_2(u,v)}{\left|F_1^*(u,v)\right|} = C_{POMF}(u,v) \quad \bullet\!\!-\!\!\circ \quad c_{POMF}(x,y) \qquad (6.16)$$

In Abbildung 6.3b ist das Ergebnis der Phasenkorrelation dargestellt. Die Abbildung zeigt anschaulich, daß das Maximum deutlich stärker und schärfer ausgeprägt ist als bei der einfachen Korrelation und damit auch besser detektierbar ist.



(a) (b) (c)

Abbildung 6.3: Ein 64×64 Pixel großer Ausschnitt aus der Rücktransformierten der im Frequenzbereich korrelierten Bilder [LEHM98]. (a) einfache Korrelation (b) Phasenkorrelation (c) symmetrische Phasenkorrelation.

Ein Nachteil der in Gleichung (6.16) dargestellten Phasenkorrelation ist, daß dieser Filter a-priori fest konfiguriert ist und nicht auch vom Eingangssignal f_2 abhängen kann. Nun kann man bei der Bildregistrierung im Frequenzbereich aber ausnutzen, daß sowohl das Referenzbild f_1 als auch das zu filternde Bild f_2 in Betrag und Phase bekannt sind. Normiert man nun das Korrelationssignal auf die Beträge der beiden Fourier-Transformierten F_1 und F_2, so bekommt man die Gleichung für die symmetrische Phasenkorrelation:

$$\frac{F_1^*(u,v)\cdot F_2(u,v)}{\left|F_1^*(u,v)\right|\cdot\left|F_2(u,v)\right|} = C_{SPOMF}(u,v) \;\bullet\!\!-\!\!\circ\; c_{SPOMF}(x,y) \qquad (6.17)$$

Mit Gleichung (6.15) läßt sich der Bruch in (6.17) vollständig kürzen und man erhält:

$$C_{SPOMF} = e^{j2\pi(u\delta_x+v\delta_y)} \;\bullet\!\!-\!\!\circ\; c_{SPOMF}(x,y) = \varphi(\delta_x,\delta_y) \qquad (6.18)$$

Dies bedeutet, daß bei der symmetrischen Phasenkorrelation das Maximum als Dirac-Stoß an der Stelle (δ_x,δ_y) im Ortsraum dargestellt wird:

$$\varphi(x,y) = \begin{cases} \infty & \text{für } x=y=0 \\ 0 & \text{sonst} \end{cases} \qquad \text{und} \quad \iint \varphi(x,y)\,dx\,dy = 1 \qquad (6.19)$$

In Abbildung 6.3c ist die symmetrische Phasenkorrelation für das hier betrachtete Beispiel dargestellt. Das Maximum ist ein deutlich ausgeprägter Peak, der sicher und leicht detektiert werden kann. Insbesondere ist $c_{SPOMF}(x,y)$ unabhängig von den Amplitudenspektren $|F_1|$ und $|F_2|$ der auszurichtenden Bilder.

6.2 Kepstrum-Filterung

Eine weitere Möglichkeit, eine relative Verschiebung zwischen zwei Bildern mit Hilfe der Fourier-Transformation zu bestimmen, ist durch die Kepstrum-Technik gegeben. Das (Power-) Kepstrum wurde im Jahr 1963 erstmals von BOGERT et al. definiert und beschrieben als eine eindimensionale heuristische Technik, um Echolaufzeiten und -amplituden in zusammengesetzten, verrauschten seismologischen Signalen zu detektieren [BOGE63]. Erst später wurde es von OPPENHEIM als ein Spezialfall der strukturerhaltenden (homomorphen) Systemtheorie erneut abgeleitet [OPPE75]. Als Power-Kepstrum $\mathsf{C}\{f(t)\}$ einer Funktion wird demnach allgemein das Power-Spektrum des Logarithmus des Power-Spektrums dieser Funktion definiert. Unter dem Power-Spektrum (oder auch Leistungsspektrum) versteht man dabei die quadrierte Betragsfunktion der Fourier-Transformierten $\mathsf{F}\{..\}$, siehe Gleichung (5.19). Diese ist nach dem Wiener-Khintchine-Theorem [WAHL89] gleich der Fourier-Transformierten der Autokorrelationsfunktion. Im Zeitkontinuierlichen gilt also:

$$\mathsf{C}\{f(t)\} = \left|\mathsf{F}\left\{\log\left|\mathsf{F}\{f(t)\}\right|^2\right\}\right|^2 \qquad (6.20)$$

Man beachte, daß dabei die zweite Fourier-Transformation nicht rückwärts ausgeführt wird, d.h. man transformiert nicht in den Ortsraum zurück wie bei der inversen Fourier-Transformation zur Bildfilterung (vgl. Abbildung 5.11). Bei der ursprünglichen Anwendung der Kepstrum-Filtrung gelangt man also durch die doppelte Fourier-Transformation vom Zeitbereich über den logarithmierten Frequenzbereich mit der Einheit *Schwingung* / *Sekunde* der Abzissenwerte in eine Art Zeitbereich mit der Einheit *Schwingung* / (*Schwingung* / *Sekunde*) = *Sekunde*. Durch Buchstaben- und Silbendrehung haben BOGERT et al. aus dem Wort „Spektrum“ das Kunstwort „Kepstrum“ und in Abgrenzung zum Ortsraum und Frequenzraum für (6.20) den Begriff Kepstral-Ebene eingeführt. Von den in Tabelle 6.1 zusammengestellten Kunstworten und Phrasen hat sich jedoch nur das Kepstrum etabliert. Durch Anwendung des Verschiebungssatzes der Fourier-Transformation (5.24) läßt sich leicht zeigen, daß das zeitverzögerte Echo eines eindimensionalen Signales im logarithmierten Power-Spektrum des zusammengesetzten Signales eine kosinusförmige Signalschwingung verursacht. Diese bewirkt ein lokales Maximum im Kepstrum. Die Echolaufzeit läßt sich dann sehr einfach durch die Position des ersten Peaks im Kepstrum ermitteln.

Spektral-Ebene	Kepstral-Ebene
Analyse	Alanyse
Filterung	Lifterung
Frequenz	Quefrenz
harmonisch	rahmonisch
Magnitude	Gamnitude
Periode	Repiode
Phase	Saphe
polar	lopar
Radius	Darius
Spektrum	Kepstrum

Tabelle 6.1 Die Kunstworte zur Unterscheidung von Spektral- und Kepstral-Ebene.

6.2.1 Zweidimensionale Kepstrum-Filterung

Die Kepstrum-Technik kann aufgrund der Separierbarkeit der zweidimensionalen Fourier-Transformation ohne Probleme in die digitale Bildverarbeitung übertragen werden. Zur Translationskorrektur geht man im allgemeinen wie folgt vor. Zunächst werden das Referenzbild $f_1(x,y)$ und das um (δ_x,δ_y) verschobene, auszurichtende Bild $f_2(x,y) = f_1(x-\delta_x, y-\delta_y)$ in einem neuen Bild, dem Kepstrumfenster, zusammengeführt:

$$w(x,y) = f_1(x,y) + f_2(x,y) - D = f_1(x,y) * \left(\varphi(x,y) + \varphi\left(x-\delta_x, y-(D+\delta_y)\right)\right) \qquad (6.21)$$

Dabei ist es zunächst unerheblich, ob die beiden Bilder der Breite M direkt aufeinander (D=0) oder nebeneinander (D=M) plaziert werden, siehe auch Kapitel 6.1.3. Die schrittweise Berechnung des Kepstrums ergibt nun:

$$\begin{aligned} w(x,y) \circ\!\!-\!\!\bullet\, W(u,v) &= F_1(u,v)\cdot\left(1+e^{-j2\pi(u\delta_x+v(D+\delta_y))}\right) \qquad (6.22)\\ |W(u,v)|^2 &= |F_1(u,v)|^2\cdot\left(2+2\cos\left(2\pi(u\delta_x+v(D+\delta_y))\right)\right)\\ \log|W(u,v)|^2 &= \log|F_1(u,v)|^2+\log(2)+\log\left(1+\cos\left(2\pi(u\delta_x+v(D+\delta_y))\right)\right) \end{aligned}$$

Zur weiteren Berechnung wird die Reihenentwicklung des Logarithmus benötigt [BRON95]:

$$\log(1+x) = \sum_{n=1}^{\infty}(-1)^{n+1}\left(\frac{x^n}{n}\right) \qquad \textit{für} \quad -1 < x \le 1 \qquad (6.23)$$

Damit folgt dann aus (6.22) weiter:

$$\begin{aligned} \log|W(u,v)|^2 &= \log|F_1(u,v)|^2+\log(2)+\sum_{n=1}^{\infty}\frac{(-1)^{n+1}}{n}\cos^n\left(2\pi(u\delta_x+v(D+\delta_y))\right) \qquad (6.24)\\ &= \log|F_1(u,v)|^2+\log(2)+\cos\left(2\pi(u\delta_x+v(D+\delta_y))\right)\\ &\quad -\frac{1}{2}\cos^2\left(2\pi(u\delta_x+v(D+\delta_y))\right)\\ &\quad \pm \ldots \end{aligned}$$

Durch eine weitere Berechnung des Fourier-Power-Spektrums gelangt man dann auf die gesuchte Kepstral-Ebene c_{KEP}. Mit den Potenzen der trigonometrischen Funktionen:

$$\cos^2(x) = \frac{1}{2}\left(1+\cos(2x)\right) \qquad (6.25)$$

$$\cos^3(x) = \frac{1}{4}\left(3+\cos(x)+\cos(3x)\right)$$

$$\cos^4(x) = \frac{1}{8}\left(3+4\cos(2x)+\cos(4x)\right)$$

...

Folgt schließlich für das Kepstrum:

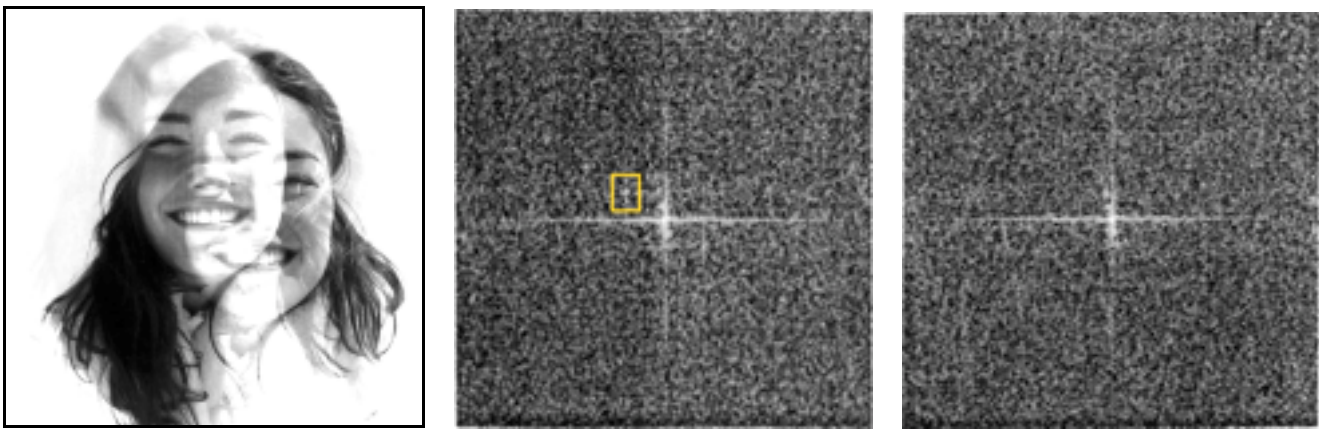
$$C\{x(x,y)\} = C\{f_1(x,y)\}+\sum_{n=1}^{\infty} A_n\varphi\left(x\pm n\delta_x, y\pm n(D+\delta_y)\right) \quad (6.26)$$

$$= C\{f_1(x,y)\}+A_0\varphi(x,y) + A_1\varphi\left(x\pm\delta_x, y\pm(D+\delta_y)\right) + A_2\varphi\left(x\pm 2\delta_x, y\pm 2(D+\delta_y)\right) + \ldots$$

Dabei nehmen die Gewichte $A_n \in$ IR der Dirac-Impulse kontinuierlich ab: $A_n > A_{n+1}\ \forall n \in \mathrm{IN}_0$. Im Fall von D=0 werden die zu registrierenden Bilder dann einfach addiert, siehe Abbildung 6.4. Aus Gleichung (6.26) ergibt sich dann für die Bewertungsfunktion c_{KEP}:

$$c_{KEP}(x,y) = C\{w(x,y)\}+C\{f_1(x,y)\} \quad (6.27)$$

$$= A_0\varphi(x,y)+A_1\varphi\left(x\pm\delta_x, y\pm\delta_y\right)+A_2\varphi\left(x\pm 2\delta_x, y\pm 2\delta_y\right)+\ldots$$



(a) **(b)** **(c)**

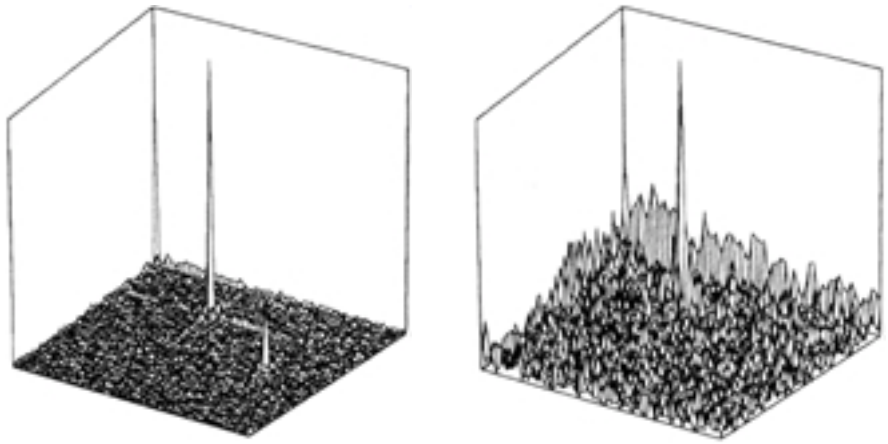
Abbildung 6.4: Durchführung der Kepstrum-Filterung c_{KEP} mit D=0. In (b) wird die Kesptralebende dargestellt, die Dirac-Stöße an den ganzahligen Vielfachen sind leider nur sehr schlecht zu erkennen. Durch Subtraktion des Kepstrums eines einzelnen Bildes (c) können die Dirac-Stößer leichter extrahiert werden.

Zur Berechnung der Gleichung (6.27) werden insgesamt 4 Fourier-Transformationen benötigt, um $c_{KEP}(x,y)$ zu berechnen. Bildet man den Quotienten zwischen Fensterfunktion und Referenzbild nach der ersten Fourier-Transformation, kann $c_{KEP}(x,y)$ mit nur 3 Fourier-Transformationen berechnet werden:

$$c_{KEP}(x,y) = \left|\mathsf{F}\left\{\log\left(\frac{|W(u,v)|^2}{|F_1(u,v)|^2}\right)\right\}\right|^2 \quad (6.28)$$

$$= A_0\varphi(x,y) + A_1\varphi\left(x \pm \delta_x, y \pm \delta_y\right) + A_2\varphi\left(x \pm 2\delta_x, y \pm 2\delta_y\right) + \ldots$$

In Abbildung 6.5a ist das Ergebnis unseres Beispiels, d.h. ein 64×64 Bildpunkte großer Ausschnitt der Kepstral-Ebene c_{KEP} , dargestellt. Wie bei der Korrelation im Frequenzbereich (vergleiche Abbildung 6.3) ist der Dirac-Stoß an der Stelle (18,12), der die Verschiebung bestimmt, deutlich erkennbar. Das Gewicht der Impulse an den ganzzahligen Vielfachen der Verschiebung nimmt schnell ab. Der Peak im Ursprung hat das höchste Gewicht in der Kepstral-Ebene, so daß die Voraussetzung (6.2), die an die Bewertungsfunktion c gestellt wurde, strenggenommen nicht erfüllt ist.



(a) **(b)**

Abbildung 6.5: Ergebnis der einfachen Kepstrum-Filterung c_{KEP} (a) und der erweiterten Kepstrum-Filterung c_{DKEP} (b) [LEMH98].

Mit dem Parameter D kann nun die Position der Peaks in der Kepstral-Ebene gesteuert werden und damit das erste Hauptmaximum mit dem Gewicht A_1 weiter vom Peak im Ursprung entfernt werden. Bezeichnet M die Breite der kontinuierlichen Bildfunktionen f_1 und f_2, so werden für $D=M$ die Bilder nebeneinander in das Kepstrum-Fenster gesetzt, siehe Abbildung 6.6. Die Peaks sind dann so weit vom Ursprung der Kepstral-Ebene entfernt, daß auch auf die gesonderte Berechnung und Subtraktion von $\mathsf{C}\{f_1(x,y)\}$ in Gleichung (6.27) verzichtet werden kann. Mit Gleichung (6.26) ergibt sich dann eine Kepstrum-Variante, für die nur noch zwei Fourier-Transformationen benötigt werden:

$$\begin{aligned} c_{DKEP}(x,y) &= \mathsf{C}\{w(x,y)\} \qquad (6.29) \\ &= \mathsf{C}\{f_1(x,y)\} + \sum_{n=1}^{\infty} A_n\varphi\left(x \pm n\delta_x, y \pm n(D_M + \delta_y)\right) \end{aligned}$$

Jedoch ist die zu transformierende Bildfunktion nun doppelt so groß. Abbildung 6.5b zeigt das Maximum der erweiterten Kepstral-Ebene. Durch die Verschiebung der Dirac-Stöße um den Betrag D_M ist nun kein störender Peak im verschobenen Ursprung vorhanden. Damit ist c_{DKEP} eine im Sinne der Definition (6.1) geeignete Bewertungsfunktion.

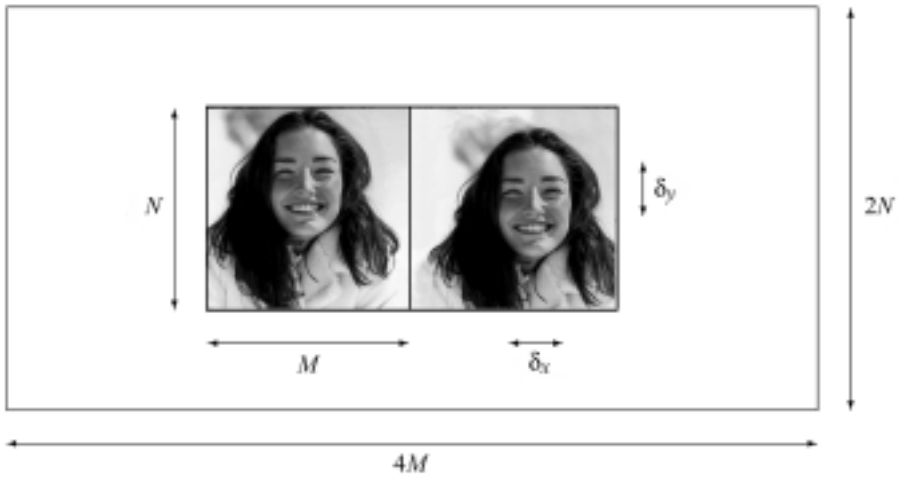


Abbildung 6.6: Anordnung der Bilder nebeneinander im Kepstrum-Fenster.

6.2.2 Diskrete Berechnung

Genau wie die Fourier-Transformation auf den diskreten Bereich übertragbar ist (siehe Kapitel 5.3.3), gilt dies auch für die Kepstrum-Filterung. Eine Besonderheit der diskreten Fourier-Transformation ist die Periodizität der diskreten Bilder, die sowohl im Orts- als auch im Frequenzbereich impliziert wird, vergleiche Definition (5.28). Daher ergeben sich die Positionen der Dirac-Stöße in der diskreten Kepstral-Ebene gegenüberseitig wiederkehrend (engl. wrap around). Peak-Positionen, die an einer Stelle die Kepstral-Ebene verlassen würden, setzen sich an der gegenüberliegenden Seite in der Kepstral-Ebene fort. Abbildung 6.7 zeigt diese Eigschaft der diskreten Kepstrum-Filterung für die erweiterte Kepstral-Ebene $c_{DEKP}(x,y)$. Für die diskreten Bildfunktionen f_1 und f_2 sei nun die Größe $M \times N$ angenommen. Die Bilder werden mit D=M mittig nebeneinander in das $(2M \times 4N)$-Kepstrum-Fenster plaziert. Die Dimensionen des Fensters sind so gewählt, daß der Peak mit dem Gewicht A_0 im markierten Suchbereich nicht auftritt. Das erleichtert die Detektion des Dirac-Stoßes mit dem Gewicht A_1, siehe auch Kapitel 7.

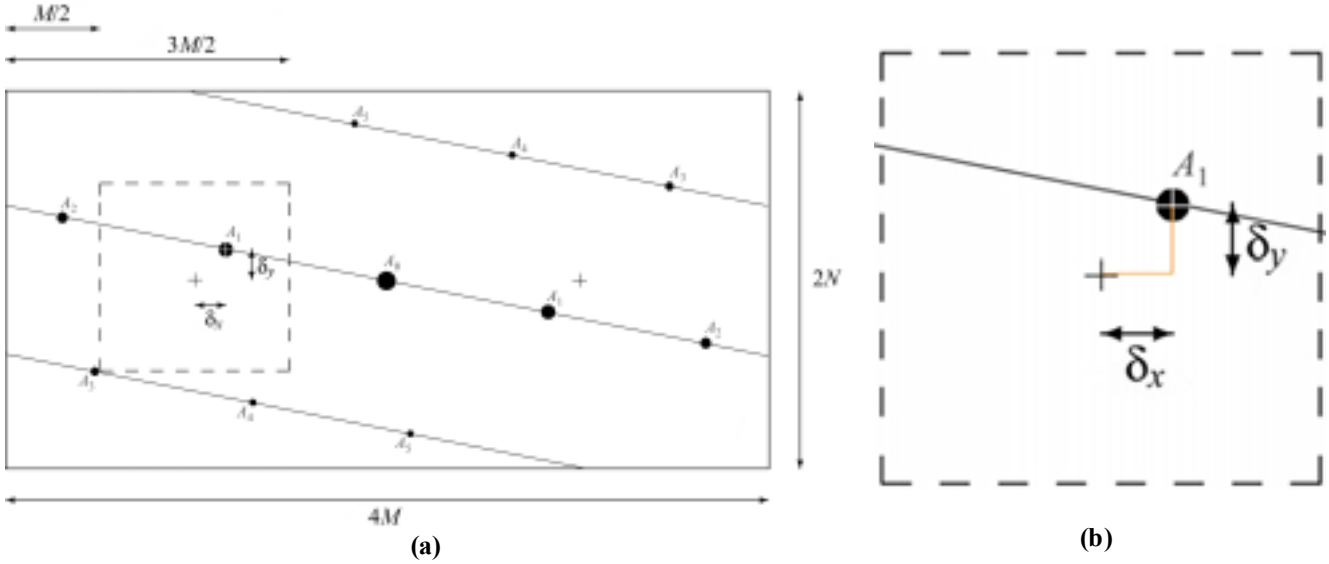


Abbildung 6.7: Wiederholung der Maxima bei der diskreten Kepstrum-Filterung. Schematische Ansicht der diskreten Kepstral-Ebene(a), vergrößerter Ausschnitt des Suchbereichs (b).

Auch bei der (symmetrischen) Phasenkorrelation wird die diskrete Berechnung durch die implizierte Wrap-Around-Situation gestört. Zwei Bilder derselben Szene, die registriert werden sollen, sind in der Regel aber nicht Wrap-Around-verschoben. Vielmehr tauchen an den Bildrändern Strukturen auf, die

im jeweils anderen Bild nicht zu finden sind. Insbesondere bei großen Verschiebungen ist die kontinuierliche Theorie der Korrelation im diskreten Äquivalent nicht mehr gültig.

Aus der Periodizität der diskreten Fourier-Transformation muß eine weitere Konsequenz gezogen werden. Verwendet man das Kepstrum $c_{DKEP}(x,y)$ nach Gleichung (6.29), so entstehen durch das „Nebeneinandersetzen“ an den Bildrändern im Kepstrum-Window starke Kontrastunterschiede. Diese stellen sich, ähnlich wie im Power-Spektrum der Fourier-Transformierten, als „Fensterkreuz“ in der Kepstral-Ebene dar, vgl. Abbildung 6.4b. Da bei $c_{DKEP}(x,y)$ das Kepstrum eines einzelnen Bildes nicht mehr berechnet und subtrahiert wird, erschweren diese Strukturen die Detektion des A_1-Peaks zusätzlich. Im nächsten Abschnitt werden daher Verbesserungsmöglichkeiten der diskreten Kepstrum-Filterung behandelt.

7 Verbesserungen der Kepstrum-Filterung

Die bisher beschriebene Kepstrum-Technik zeichnet sich vor allem durch ihre Unempfindlichkeit gegenüber Rauschen aus, da die rauschempfindlichere Phaseninformation nicht direkt benutzt wird. Ebenso ist es von Vorteil, daß kein a-priori-Wissen über Bildinhalte oder Verschiebung nötig ist. Für $c_{DKEP}(x,y)$ ist die maximale noch zu detektierende Verschiebung $\left(\pm\frac{M}{2},\pm\frac{N}{2}\right)$. Durch einige Vorverarbeitungsschritte kann die Zuverlässigkeit des Verfahrens deutlich verbessert werden. Die in den Kapiteln 7.1 bis 7.4 beschriebenen Schritte zielen darauf hin, Störeinflüsse durch numerische Ungenauigkeiten zu reduzieren. Das Hintergrundrauschen der Kepstral-Ebene kann dadurch deutlich reduziert werden. Dies vereinfacht die Detektion des die Verschiebung kodierenden Peaks. Ebenso wird die Instabilität der Translationsbestimmung gegenüber Rotationsdifferenzen reduziert. Normalerweise ist in Testbildern mit mehr als zwei Grad Drehwinkeldifferenz kein Peak mehr detektiertbar [GOER95].

Eine direkte Anwendung der Technik auf die Turbo-Flash-Nierensequenzen ist aber auch durch Vorverarbeitungsschritte i.A. nicht möglich. In den Kapiteln 7.6 und 7.7 werden notwendige Anpassungen beschrieben, damit der in Kapitel 4.3.2 vorgestellte Algorithmus anwendbar ist. Insbesondere muß eine seitengetrennte Bearbeitung sowie eine Bereichsbeschränkung der Eingabebilder beachtet werden.

7.1 Vorverarbeitung

Als Vorverarbeitung beider Bilder wurde von LEE et al. eine Kantenextraktion durch Sobel-Filterung vorgeschlagen, siehe Kapitel 5.2.4. Der Sobel-Operator schwächt dabei störende Texturen ab und verstärkt die für die Kepstrum-Filterung wichtigen dominanten Bildstrukturen wie Ecken und Kanten [LEE89]. Da die dieser Arbeit zugrundeliegenden MRT-Turbo-Flash-Aufnahmen relativ stark verrauscht sind, wird zusätzlich vor der Sobel-Filterung eine Median-Filterung vorgenommen, siehe Kapitel 5.2.3. Die Empfindlichkeit der Translationskorrektur gegenüber rotierten Bildern kann dadurch auf bis zu ±4 Grad akzeptierter Winkeldifferenz gesteigert werden. Eine Erweiterung der Kepstrum-Filterung für rotierte Bilder schlägt LEHMANN vor [LEHM98]. Dabei wird mittels einer Polarkoordinatendarstellung der Bilder zunächst die Rotation korrigiert. Anschließend führt man die Kepstrum-Filterung durch. Zusätzlich wird die Möglichkeit berücksichtigt, skalierte Bilder zu korrigieren. Daraus wird eine RST-Invariante (**R**otation-**S**kalierung-**T**ranslation) zur Registrierung skalierter und rotierter Bilder entwickelt. Die hier behandelten Sequenzen weisen diese Kriterien aber nicht auf. Rotationen kommen lediglich in geringem Umfang vor; diese liegen aber ausschließlich im Toleranzbereich von ±4 Grad.

7.2 Fensterung

Eine weitere Verbesserung der in der Literatur beschriebenen Standardverfahren kann mit einer Fensterung erzielt werden, die den Leakage-Effekt abschwächt [KAMM92]. Dieser Effekt resultiert aus der Tatsache, daß für die diskrete Fourier-Transformation das reale, ortsbegrenzte Bild als periodisches, unendlich ausgedehntes Signal aufgefaßt wird (siehe Kapitel 5.3.3). Die Nahtstellen, an denen die einen Begrenzungen des Bildes an den anderen Bildrändern ansetzen, sind in der Regel unstetig und verfälschen das Spektrum. Das Ziel der Fensterung besteht darin, so viele Ableitungen wie möglich an den Signalbegrenzungen einander gleich oder einfacher gleich null zu setzen. Gleichzeitig ist aber zu beachten, daß dabei das Spektrum des Eingangssignales nicht übermäßig stark geglättet wird. Nach diesen Kriterien liefert das Kaiser-Bessel-Fenster die besten Ergebnisse [HARR78]. Die kontinuierlich angegebene Funktion kann für jede Bildzeile oder -spalte der Dimension N durch das folgende 4-Koeffizienten Kaiser-Bessel-Fenster angenähert werden:

$$w_{KB}(n) = a_0 - a_1 \cdot \cos\left(\frac{2\pi}{N} n\right) + a_2 \cdot \cos\left(\frac{2\pi}{N} 2n\right) - a_3 \cdot \cos\left(\frac{2\pi}{N} 3n\right) \quad \text{mit } \{0 \le n \le N-1\} \qquad (7.1)$$

$$\text{und} \quad a_0 = 0;40243,\ a_1 = 0;49804,\ a_2 = 0;09831,\ a_3 = 0;00122$$

7.3 Zeropadding

Die direkte Implementierung der diskreten Fourier-Transformation erlaubt Eingabebilder beliebiger Dimension, ist aber numerisch sehr aufwendig. Für ein $M \times N$-Bild wären $(M \cdot N)^2$ komplexe Additionen und Multiplikationen notwendig. In der Implementierung der Fourier-Transformation als Fast-Fourier-Transformation (FFT) werden nur noch $M \cdot N \cdot (\frac{M+N}{2})$ komplexe Operationen benötigt, jedoch sind dann nur Eingabebilder der Dimension 2^n, $n \in \mathbb{N}^+$ zugelassen [WAHL89]. Um auch andere Bilddimensionen verarbeiten zu können, muß vor der Transformation ein sog. Zeropadding durchgeführt werden, also ein „Auffüllen“ mit Nullen bis zur nächsten Potenz von Zwei. Damit hat man nun alle (nötigen) Schritte und Operatoren zur Verfügung, um die diskrete Kepstrum-Filterung durchzuführen. Das Blockschaltbild in Abbildung 7.1 zeigt die (verbesserte) Translationskorrektur mit der diskreten Kepstrum-Filterung für ein Referenzbild $f_1(x,y)$ und das zu filternde Bild $f_2(x,y)$.

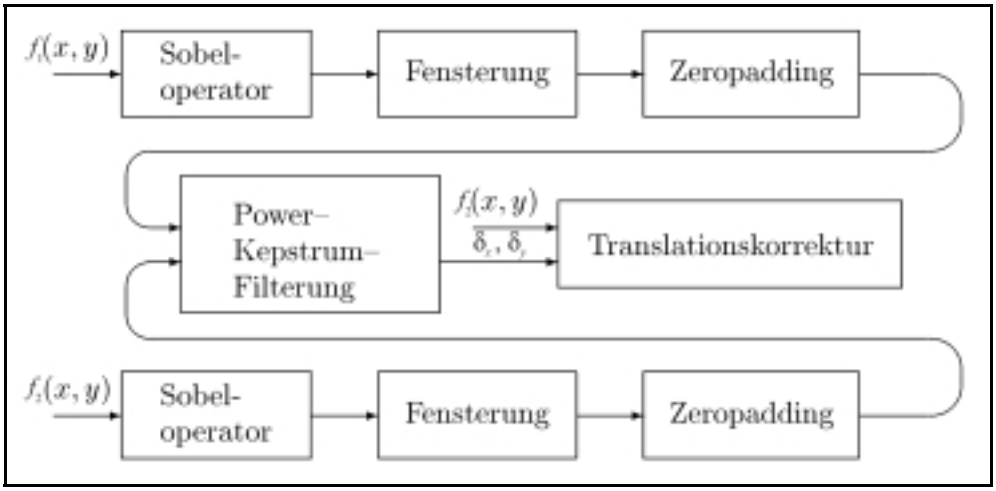


Abbildung 7.1: Prinzip der Kepstrum-Filterung [LEHM98].

7.4 Histogrammangleich

Die Anflutung des Kontrastmittels im Nierengewebe bewirkt oft einen starken Intensitätsanstieg innerhalb weniger Bilder der Sequenz. Ist dabei das Folgebild f_2 wesentlich heller und kontrastreicher als das Referenzbild f_1, so führt dies zu divergenten Ergebnissen bei der Sobel-Filterung. Die größeren Intensitätsunterschiede an den Nierenkonturen erzeugen breitere Kanten im Sobel-Bild. Zusätzlich ergeben sich neue Kanten innerhalb der Rinde / Mark bzw. Mark / Becken-Gewebegrenze, die im Referenzbild nicht existieren. Diese Eigenschaften der Bildsequenz verursachen Probleme während der Kepstrum-Filterung, die sich einerseits in numerischen Ungenauigkeiten auswirken, andererseits eine Laufzeit-Qualitätskontrolle erschweren (siehe Kapitel 6.2.1 und 4.3.2).

Deswegen wird versucht, das Folgebild dem Referenzbild durch einen Histogrammangleich in Helligkeit und Kontrast anzupassen. Zunächst werden die Histogramme beider Bilder gemäß Gleichung (5.3) erstellt und die minimalen und maximalen Werte h_{min} und h_{max} sowie die Breite h_w bestimmt:

$$h_{min} = h(i) \neq 0 \quad \text{mit } 0 \leq i < G \quad \text{und } i \text{ minimal}$$

$$h_{max} = h(i) \neq 0 \quad \text{mit } 0 \leq i < G \quad \text{und } i \text{ maximal}$$

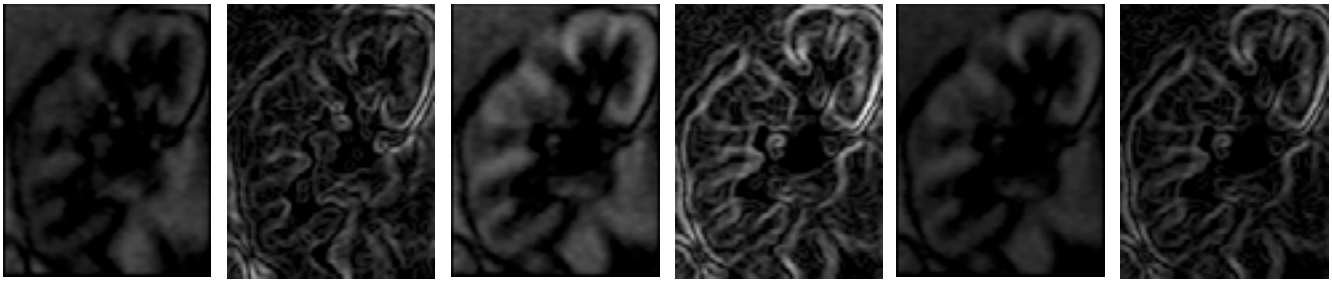
$$h_w = h_{max} - h_{min}$$

Nun wird der Histogrammangleich hergestellt, indem die Grauwerte des Folgebildes in zwei Schritten modifiziert wird. Zunächst wird ein (i.d.R negativer) Betrag $c_L = h_{f_1 min} - h_{f_2 min}$ addiert, danach wird der Grauwert entsprechend der gewünschten neuen Breite des Histogramms skaliert:

$$1) \quad f_2(m,n) = f_2(m,n) + c_L$$

$$2) \quad f_2(m,n) = h_{f_1 min} + \frac{f_2(m,n) - h_{f_2 min}}{h_{f_2 w}} \cdot h_{f1 w}$$

Das so modifizierte Folgebild kann nun mit dem Sobel-Filter weiterverarbeitet werden, ohne die Ergebnisse der Kepstrum-Filterung zu beeinflussen. Abbildung 7.2 zeigt ein Paxisbeispiel. In (a) ist das Referenzbild f_1 dargestellt, in (c) das unmodifizierte Folgebild f_2. Die sobel-gefilterten Bilder (b) und (d) weisen deutliche unterschiede auf. Abbildung 7.2e zeigt das Folgebild nach dem Histogrammangleich sowie das sobel-gefilterte Pendant (f).



(a) **(b)** **(c)** **(d)** **(e)** **(f)**

Abbildung 7.2: Beispiel zum Histogrammangleich. Referenzbild (a), Folgebild (c), die Ergebnisse deren Sobel-Filterung in (b) und (d). In (e) ist das durch einen Histogrammangleich veränderte Folgebild zu sehen. Der Unterschied zwischen (f) und (b) ist deutlich geringer als zwischen (d) und (b).

7.5 Optimierung durch die Hartley-Transformation

Die diskrete Hartley-Transformation (DHT) stellt eine Abwandlung der diskreten Fourier-Transformation dar und kann bei der Kepstrum-Bildung zu einer deutlichen Vereinfachung der Rechenoperationen und damit zur Geschwindigkeitssteigerung verwendet werden. Die DHT basiert auf einer korrespondierenden Integral-Transformation, die erstmals im Jahre 1942 von HARTLEY als Summe von doppelseitigen Sinus- und Kosinus-Transformationen beschrieben worden ist [HART42]. Eine Beschreibung des diskreten Analogons sowie die Entwicklung eines Algorithmus für eine schnelle Hartley-Transformation (FHT) wurde erstmalig von BRACEWELL angegeben [BRAC84]. Eine Zusammenfassung von Definitionen, Theoremen und Wechselbeziehungen der zwei-dimensionalen DHT und DFT ist von WATSON und POIRSON verfaßt worden. [WATS86]. Ausgelegt für die Transformation von reellen Datensätzen beliebiger Dimension werden bei der Berechnung der DHT reelle anstelle komplexer Multiplikationen und Datentypen gebraucht. Da eine komplexe Multiplikation vier reelle Multiplikationen umfaßt, sind bei der Kepstrum-Bildung, auch unter Beachtung der im Vergleich zur DFT ungünstigeren Symmetrieeigenschaften der DHT, Rechenzeitreduktionen von bis zu 50% theoretisch denkbar. Verkürzungen um bis zu 40% wurden bereits dokumentiert [STEC89]. Zusätzlich wird im Vergleich zur komplexen DFT nur der halbe Speicherplatz benötigt.

Die (separierte) diskrete Hartley-Transformierte einer zweidimensionalen reellen $M\times N$ -Bildmatrix eines Bildes $h(x,y)$ ist als ebenfalls reelle zweidimensionale $M\times N$-Bildmatrix $H_s(u,v)$ wie folgt definiert:

$$H_s(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{M-1} h(x,y)\cdot \mathrm{cas}(2\pi ux)\cdot \mathrm{cas}(2\pi vy) \qquad (7.2)$$

Die Abkürzung $\mathrm{cas}(\theta) = \cos(\theta) + \sin(\theta)$ ist dabei von HARTLEY selbst eingeführt worden. Die zweidimensionale Rücktransformierte ist gegeben durch:

$$h(x,y) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{M-1} H_s(u,v)\cdot \mathrm{cas}(2\pi ux)\cdot \mathrm{cas}(2\pi vy) \qquad (7.3)$$

Mit Gleichung (7.3) gilt die folgende Gleichung zur Berechnung des Power-Spektrums:

$$|F(u,v)|^2 = \tfrac{1}{4}\left(H_s(-u,v)+H_s(u,-v)\right)^2 + \tfrac{1}{4}\left(H_s(u,v)-H_s(-u,-v)\right)^2 \qquad (7.4)$$

Dabei wird durch zweimalige Anwendung und Logarithmierung ebenso eine Berechnung des Power-Kepstrums ermöglicht. PENGLIN, YUANLONG und DEQUN beschreiben eine heuristische Technik, um nach nur einer Hartley-Transformation eine dem Power-Kepstrum sehr ähnliche Funktion zu erzeugen [PENG91]. Demzufolge ist auf diesem Wege eine weitere deutliche Reduktion der Rechenzeiten bei der Kepstrum-Filterung möglich.

7.6 Steigerung der Genauigkeit

Eine weitere Verbesserung des Ergebnisses der Kepstrum-Filterung ist durch ein Hochskalieren der Bilder vor Anwendung der Kepstrum-Filterung möglich. Mit einem sog. „Resampling“ wird dabei sowohl für das Referenzbild als auch das verschobene Bild die Auflösung, d.h. die Anzahl der Pixel erhöht:

$$f(m,n) \rightarrow f_S(k,l) \quad mit \begin{cases} 0 \le k < K; \; K = \lfloor \tau_m \cdot M \rfloor \\ 0 \le l < L, \quad L = \lfloor \tau_n \cdot N \rfloor \end{cases} \qquad (7.5)$$

$$und \;\; \tau_m, \tau_n \in \mathbb{R} \quad \tau_m > 1, \tau_n > 1$$

Die durch die Kepstrum-Filterung berechneten Verschiebungswerte (ρ_x,ρ_y) für die skalierten Bilder müssen dann vor der abschließenden Translationskorrektur durch die Skalierungsvariablen τ_m und τ_n dividiert werden:

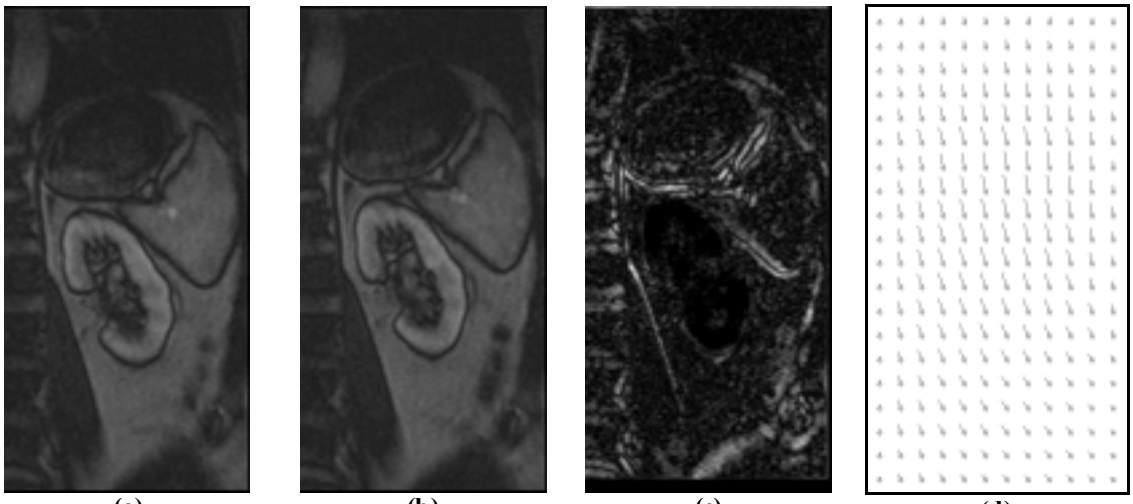
$$\delta_x = \frac{\rho_x}{\tau_m}, \;\; \delta_y = \frac{\rho_y}{\tau_n} \qquad mit \; \delta_x, \delta_y \in \mathbb{R} \qquad (7.6)$$

Das resultierende Ergebnis enthält nun keine ganzzahligen (also dem Pixelraster konformen) Verschiebungswerte (δ_x,δ_y) mehr. Algorithmisch entstehen dabei gleich zwei Probleme. Erstens darf der Bildinhalt der beiden Bilder beim Hochskalieren nicht verfälscht werden. Insbesondere ist wünschenswert, daß mittlerer Grauwert und Standardabweichung beibehalten werden: $g_{av} = g_{Sav}$ und $g_{dev} = g_{Sdev}$. Zweitens ist die Translationskorrektur um reelle Verschiebungswerte nicht ohne weiteres durchführbar, da die Grauwerte diskreter Bilder nur an ganzzahligen Koordinaten definiert sind, siehe (5.2). Beide Probleme können durch die im folgenden Kapitel behandelten Interpolationsverfahren gelöst werden.

7.7 Inhomogene Verschiebung der Bildinhalte

Die Anwendung des Kepstrum-Filters als Registrierungsverfahren für die Serienaufnahmen der Nieren hat zunächst nicht zu befriedigenden Ergebnissen geführt. Die Ursachen hierfür konnten aber recht schnell ermittelt werden. Da die Anordnung der Organe im Bauchraum (mit Ausnahme der Nieren) weitgehend asymmetrisch ist, findet eine unterschiedlich starke Verschiebung von linker und rechter Niere statt. Wie bereits in Kapitel 2 erwähnt, liegt die rechte Niere aufgrund des Platzbedarfs der darüber liegenden Leber etwas tiefer und bewegt sich meist weniger. Die Eingabe des gesamten Bildes in die Kepstrum-Filterung erzeugte daher falsche Ergebnisse. Zunächst wurde daher eine seitengetrennte Auswertung durchgeführt, indem die Sequenz einfach in der vertikal in der Mittel (im Fall der bearbeiteten Sequenzen bei x=128) getrennt wurden. Daraus resultiert die Anforderung an die Aufnahmegeometrie, daß die Wirbelsäule ungefähr in der Mitte des Bildes verlaufen sollte. Dies wurde aber ohnehin von jeder Nierenserie erfüllt.

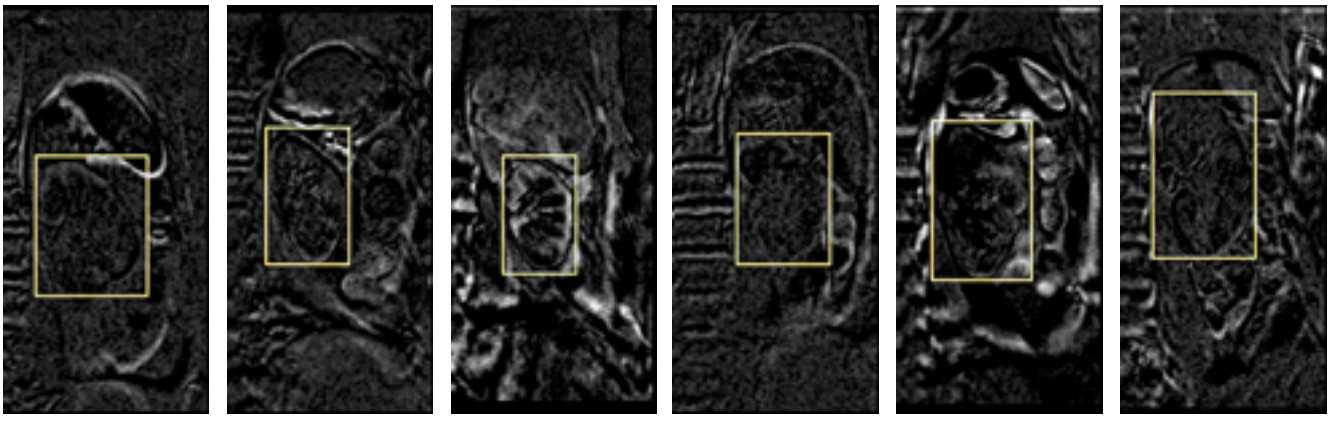
Weiterhin führt die Atmung der Patienten, bedingt durch die Anatomie des menschlichen Körpers, zu inhomogenen Verschiebungen der Bildinhalte von Bild zu Bild. Die Wirbelsäule im mittleren Bildbereich sowie Teile des Beckens im unteren Bildbereich bewegen sich beim Atemvorgang nicht mit. Anatomische Strukturen, die direkt daran grenzen, bewegen sich weniger als solche, die relativ frei im Abdomen liegen. Abbildung 7.3 soll diesen Sachverhalt verdeutlichen. Dargestellt sind zwei Folgebilder einer Serie (a) und (b). Das Folgebild wurde von Hand so korrigiert, daß nur die Nieren möglichst optimal mit dem Referenzbild zur Deckung kommen, der restliche Bildinhalt war nicht von Interesse. Abbildung 7.3c zeigt das Differenzbild der Korrektur, wobei der Bereich der Nieren mit jeweils minimalen Differenzen durch einen Rahmen markiert ist. Zur Verdeutlichung der Abbildung wurde der Kontrast des Differenzbild in (c) um 75% angehoben.



(a) (b) (c) (d)

Abbildung 7.3: Beispiel zu inhomogenen Gewebeverschiebungen.

Die inhomogenen Gewebeverschiebungen im übrigen Abdomenbereich sind deutlich zu erkennen. Abbildung 7.3d soll die Verschiebungen zusätzlich schematisch verdeutlichen, jeder 10. Bildpunkt ist mit seinem Verschiebungsvektor dargestellt. Da die Kepstrum-Filterung zunächst auf komplette Halbbilder angewendet wurde, führte dies zu numerischen Problemen. Voraussetzung (6.1) gilt nicht mehr für alle Koordinaten (x,y) der Bilder $f_1(x,y)$ und $f_2(x,y)$. Korrigiert man die Translation der Bilder von Hand, so daß die Nieren optimal zur Deckung kommen, ergeben sich hohe Standardabweichungen g_{dev} im Differenzbild. Tabelle 7.1 und Abbildung 7.4 enthalten Werte aus sechs Beispielserien für die linke Niere, zusätzlich sind die Standardabweichung und der mittlere Grauwert g_{av} nur für den markierten Bereich rund um die Niere angegeben. Zur Verdeutlichung sind die Bilder wie oben kontrastverstärkt. Die Beispiele (c) und (e) wurden bewußt gewählt, um die Problematik zu veranschaulichen. Die Standardabweichung innerhalb der ROI liegt hier deutlich über 10.



(a) (b) (c) (d) (e) (f)

Abbildung 7.4: Nichtlineare Gewebeverschiebungen, Differenzbilder von 6 Beispielserien (a)-(f).

Serie	Patient	g_{dev} Halbbild	g_{av} Halbbild	g_{dev} Selektion	g_{av} Selektion
(a)	AN370601	18,97	27,48	8,09	26,24
(b)	DL250620	18,31	25,97	10,19	25,13
(c)	HE190511	20,10	26,86	14,81	30,81
(d)	HO291015	17,96	26,83	7,21	25,39
(e)	LI280601	22,40	21,70	13,28	19,90
(f)	AM321122	20,08	30,65	7,68	28,53

Tabelle 7.1: Mittlerer Grauwert (g_{av}) und Standardabweichung (g_{dev}) der Beispielbilder aus Abbildung 7.4.

Die Detektion eines eindeutigen Maximums in Gleichung (6.5) zur Berechnung der Relativverschiebung ist dann nicht mehr möglich. Auch weist die Phase des Korrelationssignales $F_1^* \cdot F_2$ nicht mehr die deutliche Periodizität auf wie im Fall der Korrelation von verschobenen Bilder mit gleichem Inhalt, siehe Abbildung 6.2b.

Abhilfe schafft eine Begrenzung auf einen kleineren, für den Kepstrum-Filter relevanten Bildbereich, entsprechend den in Abbildung 7.4 markierten ROIs. Die Verschiebung der Pixel innerhalb dieser ROI ist in den meisten Aufnahmen fast homogen, vgl. Abbildung 7.3d. Dadurch wird eine Steigerung der numerischen Genauigkeit und der Zuverlässigkeit der Kepstrum-Filterung erreicht. Im optimalen Fall stellt sich der gesuchte Peak in der Kepstral-Ebene dann so deutlich und unverrauscht dar wie in Abbildung 7.5 gezeigt. Zusätzlich sinkt der numerische Aufwand beachtlich, da die Dimension des Kepstrumfensters (Abbildung 6.6) um ein bis zwei Zweierpotenzen reduziert werden kann.

Abbildung 7.5 zeigt einen Ausschnitt aus der Kepstral-Ebene nach Anwendung aller nun diskutierten Vorverarbeitungsschritte und Verbesserungen.

Abbildung 7.5: Peak in der Kepstral-Ebene nach Vorverarbeitung [LEHM98].

7.7.1 Nachjustierung

Zur Ergebnisüberprüfung der erweiterten Kepstrum-Filterung $c_{DKEP}(x,y)$ wird eine Nachjustierung durchgeführt. Ausgangspunkt dabei sind die auf den diskreten Bilddaten gewählten Bildausschnitte um die Nieren von Referenz- und Folgebild, die auch in die Kepstrum-Filterung eingehen. Die berechnete Verschiebung (δ_x,δ_y) wird zunächst als gültig angenommen und das Folgebild entsprechend justiert. Nun wird ein möglicher Fehler ε angenommen, der ähnlich wie bei der Filterung im Ortsbereich ein (rechteckiges) Fenster $W_\varepsilon(\delta_x,\delta_y)$ um den Verschiebungspunkt definiert:

$$W_\varepsilon(\delta_x,\delta_y) = \left\{ (\delta_i,\delta_j) : \delta_x - \varepsilon \le \delta_i < \delta_x + \varepsilon \cap \delta_y - \varepsilon \le \delta_j < \delta_y + \varepsilon \right\} \qquad (7.7)$$

Jede Position innerhalb des Fensters wird als mögliche Verschiebung angenommen, die Korrektur des Folgebildes simuliert und der mittlere Grauwert g_{av} des Differenzbildes von Referenz- und Folgebild für den Bildausschnitt um die Nieren berechnet. Durch eine Minimumsuche im Fenster W_ε kann nun die optimale Deckung der Nieren gefunden werden und bei einer Abweichung vom Ergebnis des Kepstrums die Translationskorrektur entsprechend angepaßt werden. Im folgenden Beispiel mit $\varepsilon = 2$ muß die Verschiebung um einen Pixel nach links korrigiert werden, $\delta_x = \delta_x - 1$.

$$W_\varepsilon = \begin{array}{|c|c|c|c|c|} \hline 15,2 & 13,6 & 16,7 & 16,3 & 16,9 \\ \hline 13,4 & 10,4 & 10,3 & 13,0 & 14,8 \\ \hline 11,4 & 9,7 & 9,9 & 10,0 & 13,9 \\ \hline 12,5 & 10,6 & 12,3 & 12,1 & 14,1 \\ \hline 14,8 & 13,8 & 14,6 & 15,6 & 17,7 \\ \hline \end{array} \qquad (7.8)$$

Um den Sachverhalt anschaulicher darzustellen, sind für die mittlere Zeile des Fensters (7.8) die Differenzbilder in Abbildung 7.6 dargestellt. Deren mittlerer Grauwert entspricht den Einträgen der Zeile. Man kann gut erkennen, daß das numerisch beste Ergebnis in Bild (b) ist

(a) **(b)** **(c)** **(d)** **(e)**

Abbildung 7.6: Die zur mittleren Zeile des Fensters aus (7.8) gehörenden Differenzbilder. Das beste Ergebnis findet sich in (b), entsprechend findet eine Nachjustierung statt.

Da während der Kepstrum-Berechnung bereits Abschätzungen der numerischen Genauigkeit des Ergebnisses (d.h. Qualität) berechnet werden können (vergleiche Kapitel 4.3.2), kann die Größe des Parameters ε dynamisch zur Laufzeit eingestellt werden. Untersuchungen an allen vorliegenden Nierenserien haben aber gezeigt, daß $\varepsilon = 4$ in allen Fällen zur Nachjustierung ausreicht.

7.7.2 Raster-Transformation

Die Genauigkeit der Translationskorrektur anhand der Bilddaten läßt sich nun nicht weiter steigern. Die Bildserie ist nun so weit bearbeitet, daß die Nieren von Bild zu Bild maximale Deckungsgleichheit haben. Weitere Probleme können dadurch entstehen, daß die Achse, entlang derer die Atmungsbewegungen der Niere stattfinden, zu weit von der Schnittachse der Aufnahme abweicht. Wünschenswert wäre es, wenn die beiden Achsen durch kurze Testläufe im Vorfeld der Untersuchung möglichst zur Deckung gebracht werden. Somit ergibt sich zusätzlich zu der in Kapitel 7.7 besprochenen Zentrierung der Wirbelsäule eine weitere Anforderung an die Aufnahmegeometrie. Der große Vorteil wäre dann, daß jedes Bild der Serie immer die identische Schicht der Niere zeigt. In der Praxis ist dies jedoch nicht immer zu erreichen. So kommt es, daß selbst bei optimalen Ergebnissen der Kepstrum-Filterung sich zwei Folgebilder nicht so justieren lassen, daß die Nieren hundertprozentig zur Deckung kommen. Die stärksten Unterschiede durch verschiedene Anschnitte ergeben sich meistens im Bereich des Fettgewebes im Nierenbecken; glücklicherweise ist dieses für die Berechnung der Analyseergebnisse uninteressant. Sollte aber die Fläche des Nierenanschnittes zwischen zwei Folgebildern unterschiedlich groß sein, ergeben sich weitere Probleme sowohl für die Kepstrum-Filterung also auch für die im nächsten Kapitel behandelte Differenzierung der Nierengewebe. Abbildung 7.7 veranschaulicht diesen Sachverhalt. In (a) liegen die Schnittachse des MRT und die Atmungsachse parallel, in (b) nicht. Daher ergeben sich im zweiten Fall unterschiedlich große Nieren von Folgebild zu Folgebild, wie schon durch die Darstellung der gemessenen Länge des Organs angedeutet wird.

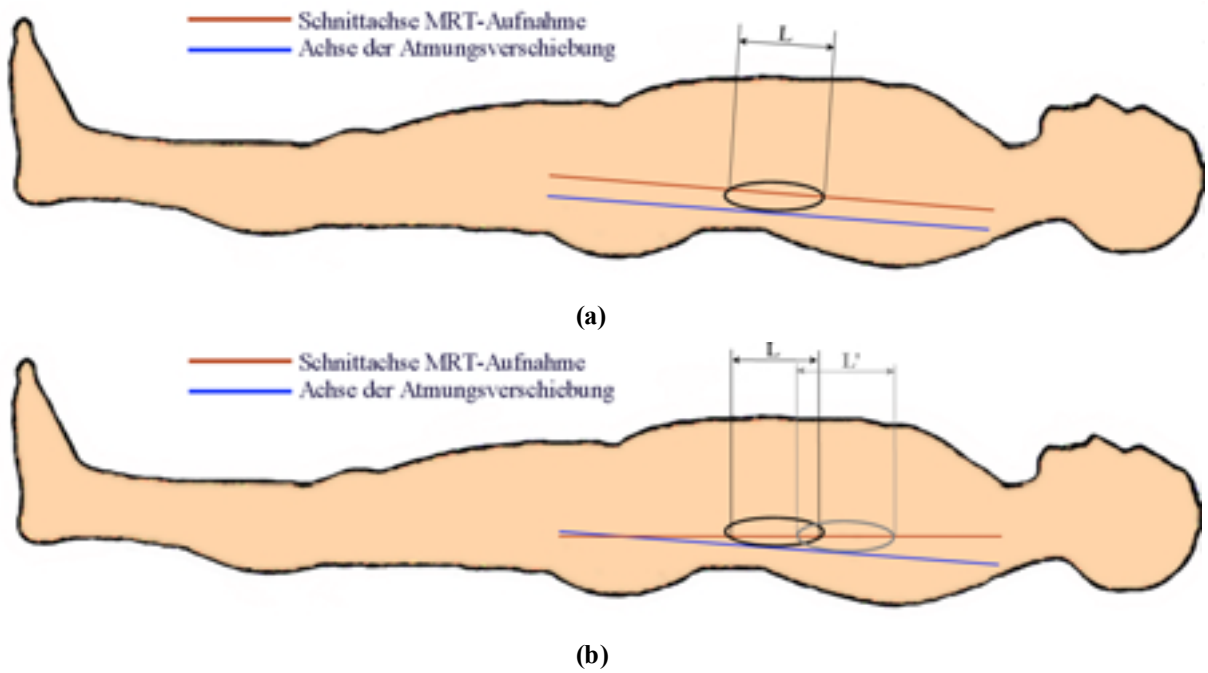


Abbildung 7.7: Unterschiedliche Anschnitte der Nieren in aufeinanderfolgenden Bildern.

Meistens liefert die Kepstrum-Filterung im Fall (b) zwar noch ein brauchbares Ergebnis, jedoch kann eine Deckungsgleichheit der Nieren nicht mehr hergestellt werden. Bleibt diese Tatsache unberücksichtigt, kann es zu sich fortsetzenden Fehlern kommen, welche letztendlich zu falschen Ergebnissen bei der Berechnung der Signal-Intensitäts-Zeit-Kurven (Kapitel 8.1) führen. In der Praxis hat sich gezeigt, daß durch die Kepstrum-Filterung entweder der obere oder der untere Bereich der Niere zur Deckung gebracht wird und die Niere damit – im ungünstigen Fall – in eine Richtung „wandert“. Abbildung 7.8 soll diese Tatsache veranschaulichen, wobei ein sich stark verändernder Nierenanschnitt für vier aufeinanderfolgende Kepstrum-Filterungen angenommen wird. Werden nun im Wechsel oberer und unterer Rand bei der Kepstrum-Filterung zur Deckung gebracht, so entsteht der beschriebene Fehler. Die „dunklen Nieren“ stellen jeweils das Referenzbild und die „hellen Nieren“ das zu korrigierende Folgebild dar. Die Ergebnisbilder der Filterung sind blau hinterlegt, wobei das gewünschte – optimale – Ergebnis der Kepstrum-Filterung durch die orangefarbigen Linien angedeutet ist.

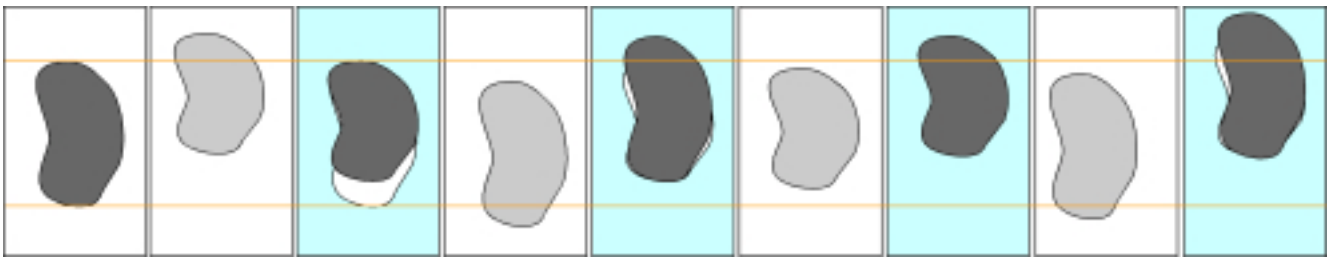


Abbildung 7.8: „Wandern“ der Nieren durch sich fortsetzende Fehler bei der Kepstrum-Filterung.

Leider ist die Lösung hier nicht so einfach wie beim zuvor beschriebenen Problem der inhomogenen Verschiebungen. Nur durch eine inhaltliche Veränderung des zu justierenden Folgebildes ist eine Deckungsgleichheit der Nieren zu erreichen. Die Veränderung realer Bildinhalte mag vom Radiologen sicher keine uneingeschränkte Zustimmung bekommen. Hier kann sie jedoch helfen, die Berechnung falscher Ergebnisse und somit ein Versagen des Gesamtsystems zu verhindern.

Zur Berechnung der Deformation des Bildrasters werden zunächst im (kepstrum-gefilterten und justierten) Folgebild f_2 der Dimension $M \times N$ eine Menge P von Paßpunkten p bestimmt und auf ihr

Übereinstimmen mit dem Referenzbild überprüft. Um diesen Schritt des Algorithmus nicht zu langsam zu machen, sollte die Anzahl der Paßpunkte nicht zu groß sein ($|P|<20$), außerdem wird ein Mindestabstand d_{min} festgelegt. Da die Paßpunkte möglichst auf dem oberen und unteren Rand der Niere liegen sollen, kann zusätzlich das Gradientenbildes g_{sob} zur Bestimmung herangezogen werden. Damit ergeben sich drei (notwendige) Kriterien für die Paßpunktbestimmung:

$$p(m,n) \in P \Leftrightarrow \begin{cases} 0 \le m < M \wedge 0 \le n < N \wedge (n < \frac{N}{4} \vee n > \frac{3N}{4}) \\ \forall p' \in P : dist(p, p') \ge d_{min} \\ g_{sob}(m,n) > \max(g_{sob}) \cdot 0{,}9 \end{cases} \qquad (7.9)$$

Um jeden Paßpunkt wird nun eine quadratische Region ausgeschnitten und als Maske (engl. template) für die Korrelation mit dem Referenzbild verwendet (engl. template matching). Abbildung 7.9 zeigt das $K \times K$ große Template $t(k,l)$ aus dem Folgebild mit $k,l = \{0,1,...,K-1\}$ und dem $U \times U$ großen Suchbereich $s(u,v)$ im Referenzbild mit $u,v = \{0,1,...,U-1\}$ und $U > K$.

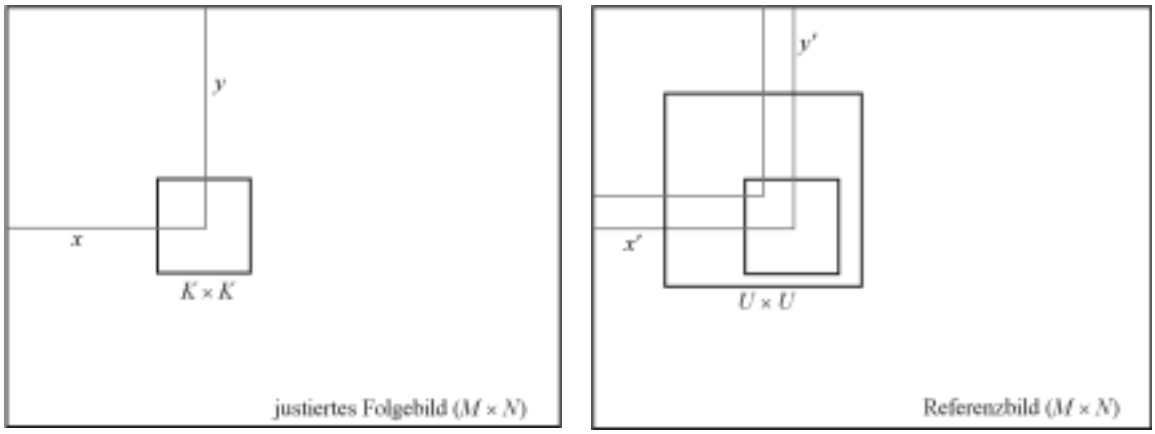


Abbildung 7.9: Kleineres Template aus dem justierten Folgebild und Suchbereich des Referenzbildes zur Bestimmung der Paßpunktkoordinaten x', y'.

Da die Größe der Templates relativ klein ist, läßt sich die Korrelation als Faltung im Ortsbereich durchführen. An der Position (x',y') ist der Betrag des Korrelationskoeffizienten (6.11)

$$c_{CCC} = \left| \frac{\sum_{u,v} \left(t(u,v) - g_{av\,t}\right)\left(s(u,v) - g_{av\,s}\right)}{\sqrt{\sum_{u,v} \left(t(u,v) - g_{av\,t}\right)^2 \sum_{u,v} \left(s(u,v) - g_{av\,s}\right)^2}} \right|$$

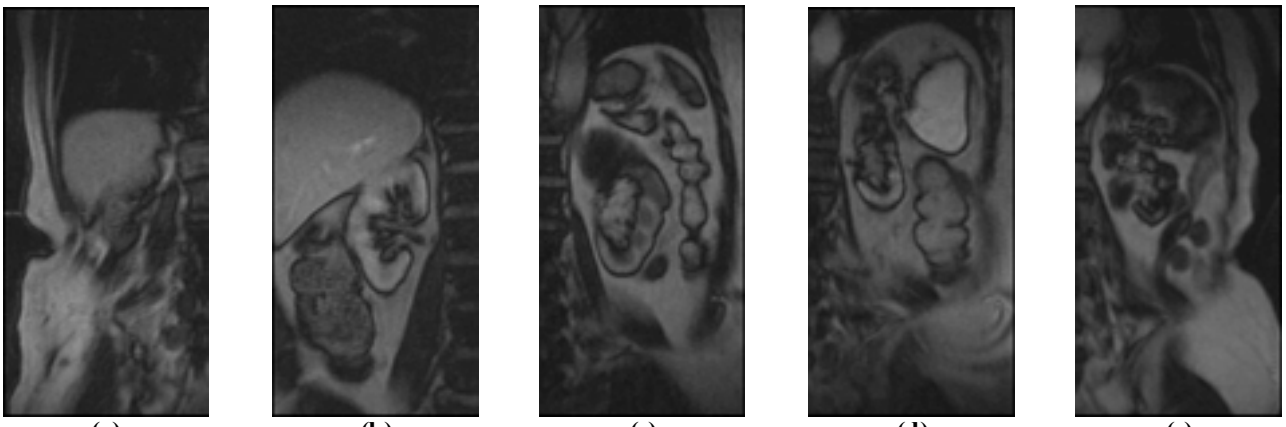
dann maximal. Die Berechnung wird für alle $p \in P$ durchgeführt und die Ergebnisse in ein leeres Deformations-Raster $\Delta(m,n) = (x',y')$, mit $0 \le m < M \wedge 0 \le n < N$ und $x', y' \in \mathbb{R}$ eingetragen. Dessen Werte sind initial Null (0,0). Damit ist die optimale Translation für alle Paßpunkte bestimmt. Für alle übrigen Positionen in Δ werden die Verschiebungen durch lineare Interpolation bestimmt. Das Deformations-Raster wird nun auf das justierte Folgebild angewendet und die Pixel des Bildes entsprechend den in Δ eingetragenen Werten verschoben. Zur Berechnung der Pixel-(Grau-)Werte an reellen Rasterkoordinaten wird wieder die im vorigen Abschnitt (5.4.2) diskutierte kubische Interpolation verwendet. Durch dieses Verfahren kann die gewünschte Deckungsgleichheit der Nieren im Folgebild wieder hergestellt werden.

7.8 Segmentierung

Wie bereits in Kapitel 7.7 diskutiert, ist eine Begrenzung des für die Translationskorrektur verwendeten Bildbereichs sinnvoll und notwendig. Wünschenswert wäre natürlich eine automatische

Detektion dieses Bereiches um die Nieren herum, wobei eine viereckige ROI völlig ausreichend wäre. Ursprünglich war gedacht, diese Auswahl durch den Benutzer vornehmen zu lassen., im Laufe der Unsetzung sollte dies dann durch eine automatische Segmentierung bewerkstelligt werden. Auch die im folgenden Kapitel behandelte Gewebedifferenzierung benötigt eine Markierung der Nieren entlang ihrer Kontur durch eine ROI, siehe Abbildung 8.1a. Wie sich während der Implementierung zahlreicher Standard-Lösungsmethoden herausstellte, war diese vermeintlich triviale Aufgaben jedoch im Zeitrahmen dieser Untersuchung nicht zu lösen. Die variable Natur der menschlichen Anatomie äußert sich in stark unterschiedlichen Bildeigenschaften, wenngleich alle Aufnahmen Nieren darstellen. Auch die bereits in Kapitel 3.2 beschriebene artifizielle Konturbetonung der Nieren ist stark unterschiedlich ausgeprägt. Die Summe an Varianzen verhindert leider die Anwendung eines für alle Serien durchführbaren Verfahrens. Ein sinnvoller Ansatz wäre sicherlich, ein auf unscharfer Logik basierendes Verfahren zu entwickeln, bei dem das Bildobjekt Niere durch unscharfe Definitionen festgelegt wird. Auch hier dürfte die große Varianz des Bildmaterials aber zu Problemen oder sehr komplexen Fuzzy-Sets führen. Lesern, denen diese Thematik bekannt ist, mögen sich Objektbeschreibungen der linke Nieren in Abbildung 7.10 überlegen.

Anatomische Unterschiede ergeben sich allein schon durch das Alter und vor allem das Gewicht der Patienten. Nierenzysten treten bei älteren Patienten mit zunehmender Häufigkeit (>30% bei >60 Jahren) auf [PSCH98]. Diese zeigen sich als dunkle Strukturen, die teilweise außerhalb der Nieren liegen und somit eine Deformierung der natürlichen Form bewirken. Die direkte Nachbarschaft anderer Organe, z.B. der Leber über der rechten Niere, zeigt sich auf einigen Aufnahmen als direkte Verbindung – oft ohne erkennbare Organgrenze. Der schon diskutierte, teilweise stark unterschiedliche Anschnitt der Niere bewirkt ebenfalls größere Konturänderungen und schwächt die Konturbetonung ab. Abbildung 7.10 zeigt aus fünf Halbbilder aus ausgewählten Nierensequenzen, die dem Leser die beschriebenen Varianzen veranschaulichen sollen.



(a) **(b)** **(c)** **(d)** **(e)**

Abbildung 7.10: (Ausgesuchte) Beispiele für problematische Nieren-Sequenzen. Bewegungsartefakte (a), verschwommene Konturgrenze zur Leber (b), Gewebedefekte und Zysten (c)-(e).

Segmentierungsversuche wurden zunächst mit zwei verschiedenen Bereichswachstumsverfahren gemacht, die jeweils schlechte Ergebnisse aufgrund sehr variabler Grauwertgrenzen lieferten [WAHL89], [HABE97]. Dies läßt vermuten, daß ähnlich arbeitende Verfahren [REVO97] oder erweiterte Verfahren wie der Watershed-Algorithmus das Problem ebenfalls nicht befriedigend lösen können. Ein Strahlverfolgungsverfahren, kombiniert mit von Snakes [KASS87] geliehenen Methoden, lieferte lediglich auf einzelnen Serien recht brauchbare Ergebnisse, vergleiche Abbildung 7.11.

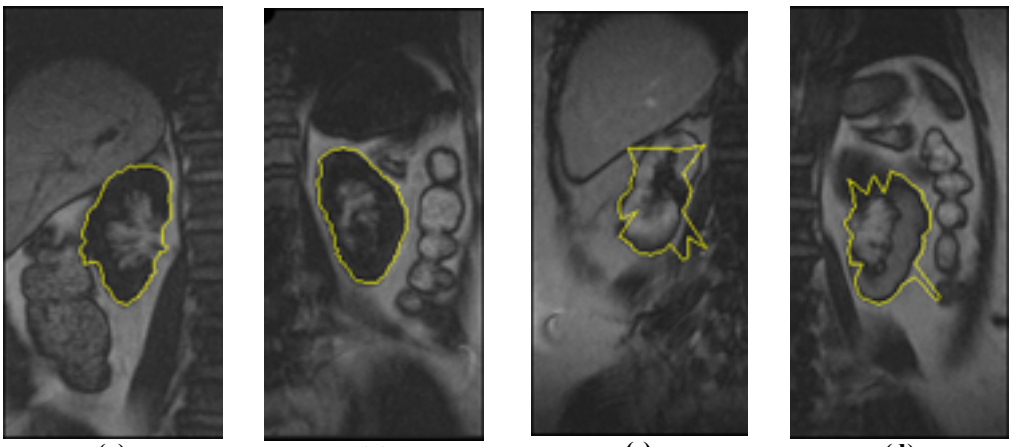


Abbildung 7.11: Ergebnisse der automatischen Detektion der Nierenkontur.

Bessere Ergebnisse sind eventuell mit einem Snake-Verfahren zu erwarten, in welches zusätzliche Informationen zur (unscharfen) Objektbeschreibung einfließen, wie z.B. Größe, Lage und Form [HILT99], [STUE99].

8 Differenzierung der Nierengewebe

Ziel der „funktionellen Perfusionsuntersuchung“ der Nieren ist die Erstellung von Signal-Intensitäts-Zeit-Kurven (engl. signal-intensity-time-graphs), kurz SIT-Kurven, sowohl für die gesamte Niere als auch gesondert für das Nierenmark und die Nierenrinde. Durch die Art des Kurvenverlaufs sowie einen Vergleich von rechter und linker Niere lassen sich dann Aussagen über über eine etwaige Organschädigung und ihren Grad machen. Auch können Rückschlüsse auf evtl. vorliegende Nierenarterienstenosen gezogen werden.

Wie schon im Kapitel 7.8 erwähnt, hat sich die Segmentierung der Nieren als schwierig herausgestellt. Dies gilt ebenso für die verschiedenen Gewebearten innerhalb des „Segments“ Niere. Im folgenden wird nun vorausgesetzt, daß die Korrektur der Atmungsverschiebung mittels Kepstrum-Filterung erfolgreich war und die Nieren sich in jedem Bild der Serie – deckungsgleich – an identischer Position befinden. Klassifizierungskriterien lassen sich dann aus der Betrachtung des SIT-Kurvenverlaufs für das Gesamtorgan ableiten. Selbst bei stark geschädigten Nieren findet sich ein charakteristischer Signalanstieg beim Einfluten des Kontrastmittels in die Niere, dessen Zeitpunkt (und damit Folgebildnummer der Serie) sich mathematisch ermitteln läßt. Abbildung 8.1 zeigt eine Beispielkurve für das Gesamtorgan (rechte / linke Niere) für die ersten 30 Bilder.

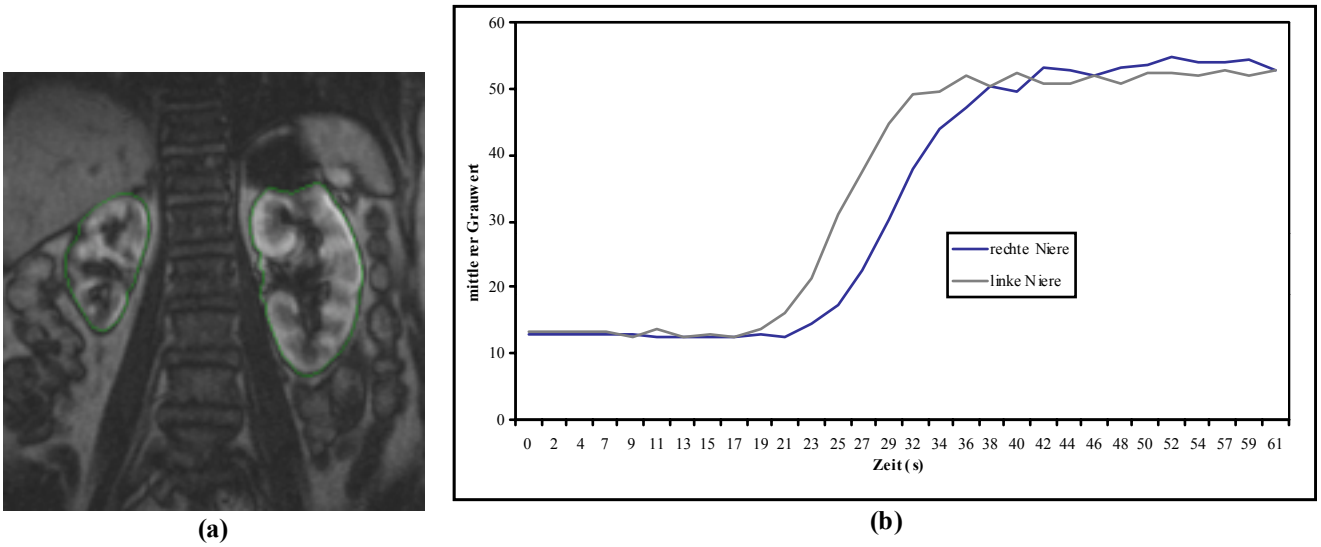


Abbildung 8.1: Markierte Gesamt-ROIs (a), Signal-Intensitäts-Zeit-Kurven der gesamten Nieren (b).

Daraus entstand schließlich die Idee, die Pixel der Niere einzeln den drei Gewebearten Nierenrinde, Nierenmark und Nierenbecken zuzuordnen, ohne dabei Segmentierungsalgorithmen verwenden zu müssen. Im folgenden wird das Vorgehen erläutert.

8.1 Berechnung der Meßkurven

Unter der Annahme einer erfolgreichen Korrektur der Atmungsbewegung ist das Erstellen der Kurve für die gesamte Niere, wie in Abbildung 8.1b gezeigt, trivial. Zunächst muß lediglich in einem Bild der Sequenz eine ROI entlang der Kontur der Niere markiert werden, wobei diese dann aufgrund der Deckungsgleichheit der Nieren auf die übrigen Bilder projiziert werden kann. Das Einzeichnen der ROI wird durch die artifizielle Konturbetonung erleichtert. Da es egal ist, in welchem Bild der Serie die ROI markiert wird, bietet es sich an, ein Bild aus dem letzten Drittel des Untersuchungsverlaufs zu nehmen. Hier ist das Nierengewebe mit Kontrastmittel angereichert und die Konturen sind deutlicher zu erkennen, siehe Abbildung 8.1a.

Für aller Leser, die nach Kapitel 4 direkt hierhin gesprungen sind, wird die verwendete Nomenklatur eventuell unklar sein. Tabelle 8.1 faßt kurz die in diesem Kapitel verwendeten Begriffe und (mathematischen) Bezeichnungen zusammen.

Bezeichnung	Schreibweise	Verweis	Erklärung
Bildpunkt	p	Kapitel 5.1	Auch (engl.) Pixel
Grauwert	g	Kapitel 5.1	Intensität des Bildpunktes
Bildfunktion	$f(m,n) = g$	Kapitel 5.1	Mathematische Beschreibung eines Rasterbildes. Der Funktionswert entspricht dem durch die ganzzahligen Koordinaten m,n Grauwert des Bildpunktes
Bildpunktmenge	$\mathcal{P}$		Darstellung einer zusammengehörenden Anzahl (Menge) von Bildpunkten durch ein Symbol
Mittlerer Grauwert	g_{av}	Kapitel 5.1.1	Aufsummieren der Intensitätswerte aller Bildpunkte einer Menge oder Bildfunktion; anschließend Teilen durch die Anzahl
Standardabweichung	g_{dev}	Kapitel 5.1.1	Aufsummieren der Differenz von Intensitätswert und mittlerem Grauwert aller Bildpunkte einer Menge oder Bildfunktion, anschließend Teilen durch die Anzahl

Tabelle 8.1: Erklärung der wichtigen Bezeichner für dises Kapitel.

8.1.1 Gesamtniere

Alle Pixel innerhalb der ROI werden dem Gesamtsegment der Niere zugeordnet, der Pixelmenge P_{ROI}. Durch Berechnung des mittleren Grauwertes g_{av} entsprechend Gleichung (5.4)

$$g_{av}(t) = \frac{1}{|\mathsf{P}_{ROI}|} \sum_{\mathsf{P}_{ROI}} f(t,m,n) \qquad (8.1)$$

sowie der Standardabweichung (5.5) g_{dev}

$$g_{dev}(t) = \frac{1}{|\mathsf{P}_{ROI}|} \sum_{\mathsf{P}_{ROI}} \left(f(t,m,n) - g_{av}(\mathsf{P}_{ROI}) \right)^2 \qquad (8.2)$$

ergeben sich die Meßkurven für die Gesamtniere. Der Einfachheit halber wird ab nun die komplette Serie als Bilderstapel (engl. stack) aufgefaßt und die einzelnen diskreten Bildfunktionen $f(m,n)$ um einen Parameter t erweitert, der den Zeitpunkt der Aufnahme angibt. Durch t wird dann die Position innerhalb der Stacks (Bildnummer) bestimmt.

8.1.2 Nierenrinde und Nierenmark

Zur Berechnung der Kurven für Nierenrinde (*cortex*) und Nierenmark (*medulla*) müssen die Pixel der Niere nun weiter unterteilt werden. Zunächst definieren man drei disjunkte Mengen entsprechend den gewünschten Gewebezuordnungen: $\mathsf{P}_{cor} \cap \mathsf{P}_{med} \cap \mathsf{P}_{pel} = \varnothing$. Die Berechnung des Schwerpunktes S_{ROI}

$$S_{ROI} = \left(\frac{sum_x}{g_{av}}, \frac{sum_y}{g_{av}}\right) \text{ mit } sum_x = \sum_{\mathsf{P}_{ROI}} m \cdot f(m,n) \quad \text{und} \quad sum_y = \sum_{\mathsf{P}_{ROI}} n \cdot f(m,n) \tag{8.3}$$

bietet die Möglichkeit, anatomische Lageinformationen auszunutzen, indem man die Nieren als annähernd ovale Struktur betrachtet. Das Nierenbecken liegt innen , gefolgt von der Markschicht und schließlich der äußeren Rinde.

Der Bereich des Nierenbeckens (*pelvis*) ist diagnostisch irrelevant und kann, da seine Pixel kein Kontrastmittel aufnehmen, gut detektiert werden. Er ist optisch dunkel und erscheint durch eine artifizielle Kontrastverstärkung im späteren Verlauf der Serie noch dunkler. Daraus ergeben sich folgende Bedingungen:

$$p_{ROI}(m,n) \in \mathsf{P}_{pel} \Leftrightarrow \begin{cases} f(t,m,n) < g_{pel} \\ f(t,m,n) \approx \text{const} \\ \text{dist}(S_{ROI},(m,n)) < d_{pel} \end{cases} \quad \text{mit } 0 \le t < t_{max} \tag{8.4}$$

Dabei beschreibt t_{max} das letzte Bild der Serie (i.d.R. 90). Dabei können die Parameter d_{pel} und g_{pel} dynamisch eingestellt werden (siehe Kapitel 9 und 10). Es hat sich gezeigt, daß sich g_{pel} gut aus den absoluten Werten der Kurve für die Gesamtniere berechnen läßt. Für d_{pel} werden prozentuale Angaben genutzt, indem von S_{ROI} ausgehend für alle Pixel von P_{ROI} der Abstand in Prozent berechnet wird. Pixel, die auf der Randkontur der Niere liegen, haben den Abstand 100.

Für die noch ausstehende Differenzierung der Pixel von Mark und Rinde kann die Tatsache ausgenutzt werden, daß im Verlauf der MRT-Aufnahme zunächst nur Pixel der Nierenrinde vermehrt Kontrastmittel aufnehmen, die Pixel des Marks hingegen nur wenig [BENN97]. Der initiale Anstieg der Kurve in Abbildung 8.1 wird daher durch einen Signalanstieg (d.h. Erhöhung des Grauwertes) der Rinden-Pixel verursacht. Aufgabe ist es nun, genau diesen Zeitpunkt t_w zu bestimmen, um anhand der ersten i Bilder der Serie ($0 \le i \le t_w$) die Unterteilung in Rinde und Mark vorzunehmen. Man betrachtet daher die Kurve zunächst als (diskrete) Funktion der Zeit: $f_{SIT}(t) = g_{av}(t)$. Diese wird durch lokale Mittelwertbildung geglättet:

$$\overline{f}_{SIT}(t) = \frac{1}{2r+1} \sum_{u=t-r}^{t+r} f_{SIT}(u) \tag{8.5}$$

Dabei reicht bereits $r = 1$ in den meisten Fällen aus. Schließlich wird die zweite Ableitung $\overline{f''}_{SIT}(t)$ auf Nullstellen untersucht, um den Wendepunkt t_w festzustellen. Im vorliegenden Fall ist der Wendepunkt bei allen Kurven mit dem Zeitpunkt des stärksten Anstiegs identisch. Algorithmisch einfacher kann daher das Maximum der ersten Ableitung festgestellt werden:

$$\overline{f'}_{SIT}(t)\Big|_{\max} = t_w \quad \text{mit } \overline{f'}_{SIT}(t) = \overline{f}_{SIT}(t) - \overline{f}_{SIT}(t-1) \tag{8.6}$$

Um nun ähnliche Klassifizierungsbedingungen wie in (8.4) auch für Mark und Rinde aufstellen zu können, lassen sich mit Hilfe von t_w weitere Parameter bestimmen. Es werden dazu Funktionen (Grauwertbilder) definiert, die Aussagen über den mittleren Grauwert, Standardabweichung und mittleren Grauwertanstieg vom Beginn der Serie bis zum Zeitpunkt t_w für alle Pixel in P_{ROI} machen.

$$g_{av(t_w)}(m,n) = \frac{1}{t_w} \sum_{z=0}^{t_w} f(z,m,n) \tag{8.7}$$

$$g_{dev(t_w)}(m,n) = \frac{1}{t_w}\sum_{z=0}^{t_w}\left(f(z,m,n) - g_{av(t_w)}(m,n)\right)^2 \quad (8.8)$$

$$g_{asc(t_w)}(m,n) = \frac{1}{t_w - 1}\sum_{z=1}^{t_w} f(z,m,n) - f(z-1,m,n) \quad (8.9)$$

Anschaulich lassen sich diese Funktionen auch als Grauwertbilder betrachten. Abbildung 8.2 stellt Gleichungen (8.7) - (8.9) sowie ein durch Bestimmung des Schwerpunktes S_{ROI} berechnetes Distanzbild dar.

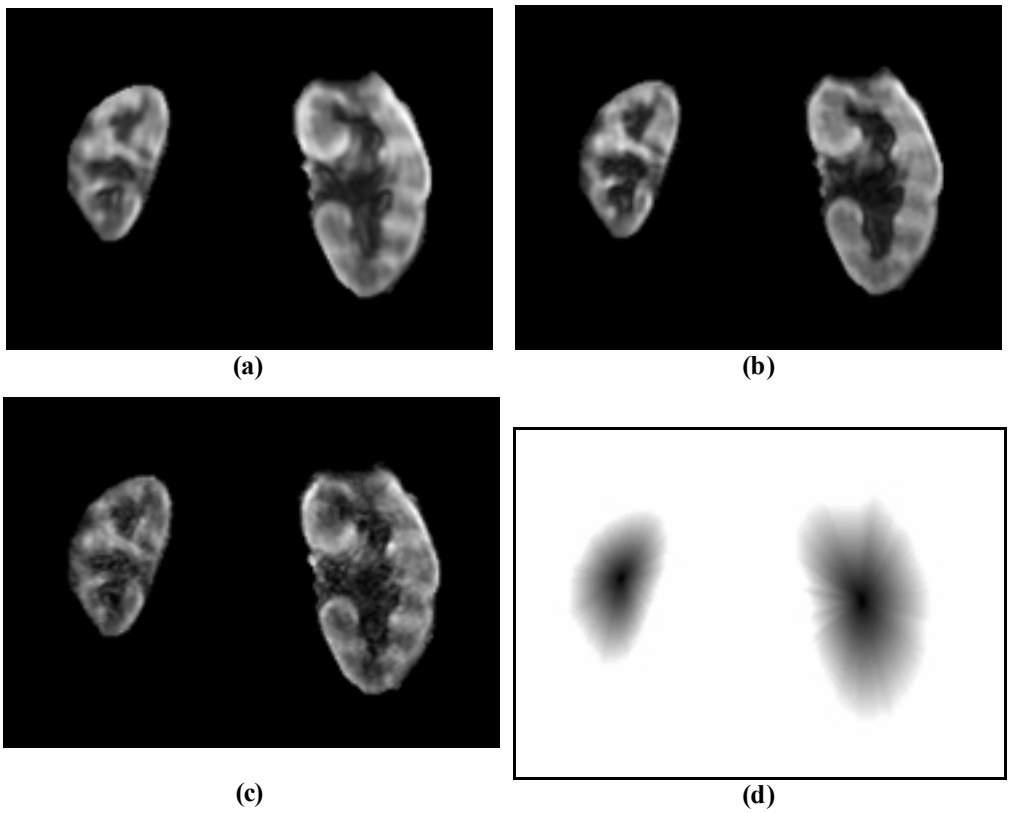


Abbildung 8.2: Darstellung der vier Hilfskriterien zur Klassifizierung (als anschauliche Grauwertbilder), berechnet anhand t_w.

Alle zum Nierenmark gehörenden Pixel lassen sich nun bestimmen durch:

$$p_{ROI}(m,n) \in \mathsf{P}_{med} \Leftrightarrow \left(p_{ROI}(m,n) \notin \mathsf{P}_{pel}\right) \wedge \begin{cases} g_{av(t_w)}(m,n) < g_{av(med)} \\ g_{dev(t_w)}(m,n) < g_{dev(med)} \\ g_{asc(t_w)}(m,n) < g_{asc(med)} \\ \mathrm{dist}(S_{ROI},(m,n)) < d_{med} \end{cases} \quad (8.10)$$

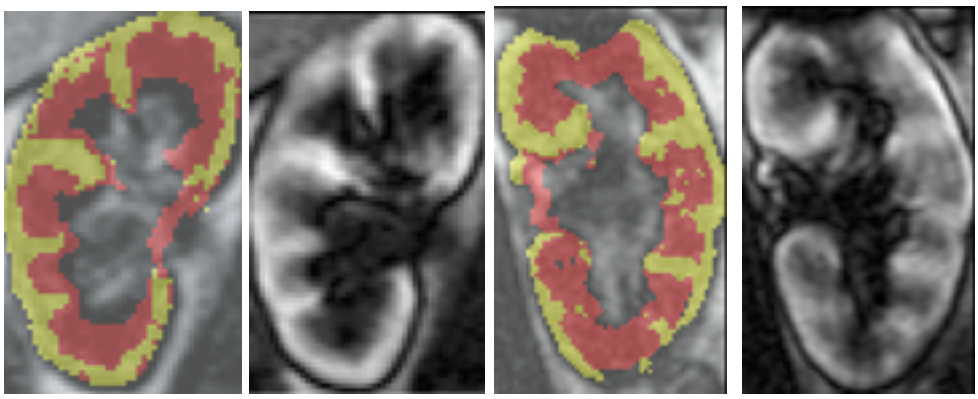
Dabei sind die vier Schwellenwerte $g_{av(med)}$, $g_{dev(med)}$, $g_{asc(med)}$ und d_{med} nicht statisch eingestellt, sondern werden durch den Verlauf der Kurve für die Gesamtniere berechnet. Außerdem sind sie, wie schon die Parameter für das Nierenbecken zuvor, dynamisch einstellbar. In Abbildung 8.3 sind die als Nierenmark klassifizierten Pixel rot markiert.

Entgegen der Erwartung wird nun keine vollständige Disjunktion der Pixel ($\mathsf{P}_{ROI} = \mathsf{P}_{cor} \cup \mathsf{P}_{med} \cup \mathsf{P}_{pel}$ und $\mathsf{P}_{cor} \cap \mathsf{P}_{med} \cap \mathsf{P}_{pel} = \varnothing$) hergestellt, indem alle übrigen Pixel automatisch als Nierenrinde klassifiziert werden. Es hat sich gezeigt, daß insbesondere bei älteren Patienten der „äußerste Ring" der Niere häufig nicht zusammen mir der Rinde einen Signalanstieg vollführt. Um dem Rechnung zu

tragen und die Ergebnisse nachträglich besser justierbar zu machen (vgl. Kapitel 10), wurden zusätzlich Schwellenwerte für die Nierenrinde eingeführt:

$$p_{ROI}(m,n) \in \mathsf{P}_{cor} \Leftrightarrow \begin{pmatrix} p_{ROI}(m,n) \notin \mathsf{P}_{pel} \\ p_{ROI}(m,n) \notin \mathsf{P}_{med} \end{pmatrix} \wedge \begin{cases} g_{av(t_w)}(m,n) > g_{av(cor)} \\ g_{dev(t_w)}(m,n) > g_{dev(cor)} \\ g_{asc(t_w)}(m,n) > g_{asc(cor)} \\ \mathrm{dist}(S_{ROI},(m,n)) > d_{cor} \end{cases} \quad (8.11)$$

Abbildung 8.3 zeigt die Pixel des *Cortex* als gelb markierte Bereiche. In (b) und (d) werden zusätzlich die Bilder der Serie dargestellt, deren Position sich durch die Berechnung von t_w ergeben hat.



(a) **(b)** **(c)** **(d)**

Abbildung 8.3: Ergebnis der Gewebeklassifizierung für die linke und rechte Niere einer Beispielserie.

8.2 Diagnose

Das System selbst nimmt keine weitere Analyse des Kurvenverlaufs vor. Diagnostisch interessant ist der Vergleich von linker und rechter Kurve für die verschiedenen Bereiche der Niere bzw. die Betrachtung des Kurvenverlaufs (Steilheit und Zeitpunkt des Anstiegs, Normalwert, Maximum, etc.). Aus einem einseitig verzögerten Anstieg können Rückschlüsse auf vorliegende Schädigungen des Nierengewebes bzw. Nierenarterienstenosen gezogen werden, vergleiche Kapitel 2. Die berechneten Werte können exportiert werden, um nach verschiedenen Kriterien statistisch ausgewertet zu werden. Sie bieten dem Radiologen wertvolle Informationen zur Diagnosestellung.

9 Programmierung

Nach den Erläuterungen der letzten beiden Kapitel zum Konzept der einzelnen Komponenten des Systems soll nun auf die Umsetzung eingegangen werden. Dabei wird nach einigen einführenden Erläuterungen zur Wahl der Programmiersprache JAVA und zum Programm IMAGEJ zunächst die Realisierung der Kepstrum-Filterung behandelt. Anschließend wird auf die Umsetzung der Klassifizierung eingegangen, um die gleiche Reihenfolge wie in den konzeptionellen Kapiteln beizubehalten.

Nach einigen allgemeinen Informationen zur Implementierung wird zunächst das Bildverarbeitungsprogramm IMAGEJ vorgestellt. IMAGEJ ist eine freie Software, die unter anderem bereits einige grundlegende Zeichenoperationen auf 8, 16 und 32-Bit-Bildern bereitstellt. Das Interessante an IMAGEJ ist, daß es sich sehr leicht um eigene Operatoren, die sogenannten *Plugins,* erweitern läßt. Daher werden die benutzten Schnittstellen von IMAGEJ erläutert, da diese in den folgenden Verarbeitungsschritten wiederholt benötigt werden.

In den folgenden Abschnitten werden der Import der Bilddaten (Nierensequenzen), die Korrektur der Atmungsverschiebung, die Berechnung der SIT-Kurven sowie der Export der Ergebnisse dargestellt. Anhand des Ablaufdiagramms in Abbildung 4.9 wird die Umsetzung des realen Systems in die Programmiersprache JAVA erklärt.

Zum Schluß des Kapitels wird noch kurz auf die Umsetzung der Segmentierung sowie der bikubischen Interpolation zur Bildskalierung eingegangen. Als Nebenprodukt ist dabei ein Skalierungs-*Plugin* für IMAGEJ entstanden, da in IMAGEJ bisher keine bikubische Interpolation zur Skalierung implementiert war.

9.1 Allgemeine Informationen zur Implementierung

9.1.1 Wahl der Programmiersprache

Der Hauptgrund für die Wahl der Programmiersprache JAVA lag im Wunsch der Anwender, ein plattformunabhängiges System zu entwickeln. Der Benutzer kann das Programm IMAGEJ (und natürlich dessen *Plugins*) unabhängig davon einsetzen, welches Betriebssystem und welche Rechnerkonfiguration er benutzt. Allerdings muß ein genügend schneller Rechner und eine *virtuelle Java-Maschine*[9] (engl. *Virtual Machine*) für das benutzte Betriebssystem vorhanden sein.

JAVA erreicht die Plattformunabhängigkeit durch den *Bytecode* als Zwischenstufe. Dieser ist plattformunabhängig und wird von der *Virtual Machine* interpretiert. Diese *Virtual Machines* sind i.d.R. für alle gängigen Plattformen verfügbar; dadurch funktioniert IMAGEJ und die renale Funktionsanalyse auf allen diesen Plattformen.

JAVA bietet aber noch viele weitere Vorteile einer modernen objektorientierten Programmiersprache. Durch die objektorientierten Spracheigenschaften können wiederverwendbare Module entwickelt werden. Im Vergleich zu C++ sind einige Eigenschaften nicht vorhanden, die dort oft nicht korrekt oder unverständlich verwendet werden. Gerade bei größeren Projekten kann die falsche Verwendung dieser Konstrukte in C++ schnell zu unübersichtlichem Quelltext führen. JAVA verwendet z.B. keine Zeigerarithmetik, sondern nur Referenzen auf Objekte. Durch die starke Typisierung und den Mechanismus der *Exceptions* wird eine hohe Robustheit erreicht.

Die Ausführungsgeschwindigkeit ist wegen des interpretierten Bytecodes zwar niedriger als für C++-Programme, aber durch JIT-Compiler[10] oder durch die Option, Maschinencode zu erzeugen, ist der Unterschied zwischen C++ und JAVA in der Ausführungsgeschwindigkeit kaum wahrnehmbar. Für spätere Erweiterungen ist auch die Ausrichtung von JAVA auf die Ausführbarkeit in Netzwerken interessant. IMAGEJ ist so konzipiert, daß es sich auch als JAVA-Applet in Browsern ausführen läßt. Somit steht einem Einsatz des Programms in komplexen Rechnernetzen nichts im Weg. Auch wäre es z.B. denkbar, die Bilddaten direkt über das Internet in das Programm einzuladen.

9.1.2 Arbeitsmittel und Arbeitsumgebung

Als Programmierwerkzeug wurde die JAVA-Entwicklungsumgebung JBUILDER 2.01 von INPRISE (BORLAND) benutzt. Diese hatte gegenüber dem JAVA Developer Kit (JDK) von SUN den Vorteil, daß ein Quelltexteditor mit Syntaxhervorhebung und ein Werkzeug zur Gestaltung der Benutzeroberflächen zur Verfügung stand. Auch das Debuggen von Programmen ist mit dem JDK sehr umständlich und wird durch den JBUILDER wesentlich komfortabler. Durch ein auf der Homepage von INPRISE zur Verfügung stehendes Update war es möglich, den JDK 1.1.8 Compiler zu benutzen und auch die original SWING-Klassen von SUN in der Version 1.1.1 zu verwenden. Dies hatte wiederum den Vorteil, daß die SWING-Bibliothek von INPRISE nicht mit dem Programm ausgeliefert werden muß, sondern das System auf jeder Plattform mit dem JDK 1.1.8 und SWING 1.1.1 von SUN lauffähig ist.

Das JDK von SUN existiert inzwischen in der Version 1.2.2, jedoch ergab sich die Notwendigkeit der Benutzung des etwas älteren Paketes 1.1.8 aus der Wahl von IMAGEJ als Grundlage zur Programmierung. IMAGEJ ist mit der *Virtual Machine* des JDK 1.1.8 schneller ausführbar. Bei höheren Versionsnummern des JDK ergeben sich verschiedene Probleme bei der Grafikdarstellung [RASB98].

Einige Experimente mit dem JDK 1.1.8 von IBM zeigten eine erheblich langsamere Ausführung der Fourier-Transformation, während sich im Bereich der Grafikdarstellung Geschwindigkeitsvorteile

[9] Die *Virtual Machine* übersetzt den vom JAVA-Compiler erzeugten Bytecode in Maschinenbefehle, die vom jeweiligen System verstanden wird. Das Programm funktioniert damit auf jedem System, für das eine *Virtual Machine* verfügbar ist.

[10] JIT: Just in Time Compiler. Mit einem JIT werden die Befehle vor der ersten Ausführung in Maschinensprache übersetzt. Da der Bytecode dann nicht bei jedem Durchlauf neu interpretiert wird, ist die Ausführung deutlich schneller.

feststellen ließen. Zu Gunsten der Gesamtrechenzeit des Systems wird daher vom IBM JDK 1.1.8 abgeraten.

Der vorliegende schriftliche Teil der Arbeit wurde mit Hilfe der Textverarbeitung WORD97 von MICROSOFT erstellt. Dazu wurden die beiden Patches SR-1 und SR-2a installiert. Auf der beiliegenden CD steht die Ausarbeitung als PDF-Datei zur Verfügung. Diese wurde über den Umweg einer Postscript-Datei mit Hilfe des Programms EXCHANGE 3.01 von Adobe erstellt.

Die Quelltextkommentierung (siehe folgendes Kapitel) erfolgte so, daß mit Hilfe des im JDK 1.2.1 von SUN enthaltenen Werkzeugs JAVADOC eine HTML-Quelltextbeschreibung erstellt werden konnte. Diese befindet sich ebenfalls auf der beiliegenden CD im Verzeichnis *JavaDoc*.

9.1.3 Quelltextkonventionen

Um den Quelltext gut lesbar und übersichtlich zu gestalten, wurden einige Konventionen für die Namensgebung, Groß- und Kleinschreibung und Kommentierung festgelegt.

Namenskonvention und Formatierung

Bezeichner werden in Englisch benannt, alle Kommentare werden in Deutsch verfaßt. Einrückungen werden jeweils im Abstand von 2 Leerzeichen vorgenommen. Nach jedem Konstruktor und jeder Methode folgt mindestens eine Leerzeile. Umlaute und Sonderzeichen werden im Quelltext grundsätzlich nicht verwendet[11].

Klassen

Klassennamen beginnen immer mit einem Großbuchstaben. Ist der Klassenname aus mehreren Worten zusammengesetzt, beginnt jedes dieser Worte mit einem Großbuchstaben. Nach der Klassendefinition erfolgt die Definition der Klassen- und Instanzvariablen. Dieser Bereich wird oft auch Kopf (engl. header) genannt. Danach werden die Konstruktoren definiert, worauf die Methoden folgen. Nachstehend ist ein Beispiel für eine Klassendefinition dargestellt.

```
class MyClass extends MyOtherClass
{
  // Klassen- und Instanzvariablen
  Object variable1;

  // Konstruktoren
  MyClass()
  {
    ..
  }

  // Methoden
  public void method1()
  {
    ..
  }
}
```

[11] Da die Klassenbeschreibungen direkt aus dem Quelltext generiert wurden, enthalten auch diese keine Umlaute und Sonderzeichen.

Methoden

Methodennamen beginnen mit einem Kleinbuchstaben. Alle folgenden Worte im Methodennamen beginnen mit einem Großbuchstaben. Methoden, die Zugriff auf eine Klassen- oder Instanzvariable ermöglichen, beginnen mit *get* bzw. *set*.

Klassen- und Instanzvariablen

Klassen- und Instanzvariablen beginnen mit einem Kleinbuchstaben. Alle folgenden Worte in deren Namen beginnen mit einem Großbuchstaben. Konstanten werden komplett in Großbuchstaben geschrieben und als *final static* deklariert.

```
// Klassen- und Instanzvariablen
int variable1;

// Konstanten
static final int constant1
```

Kommentare

Besonders großer Wert wurde auf die Kommentierung unseres Quelltextes gelegt. Jede Klasse, jeder Konstruktor, jede Methode und jede Klassen- und Instanzvariable ist so kommentiert, daß mit Hilfe des im JDK 1.2.1 von SUN enthaltenen Werkzeugs JAVADOC automatisch ein HTML-Dokument generiert werden kann, welches alle wichtigen Implementierungsinformationen enthält. Jeder dieser Kommentare beginnt mit /** und endet mit */. Mehrzeilige Kommentare werden wie im folgenden dargestellt realisiert.

```
/**
 * Diese Instanzvariable ist eigentlich nicht von Bedeutung.
 * Es kommt in diesem Fall vielmehr auf den Kommentar an.
 */
int variable1;
```

Bei Konstruktoren und Methoden ist neben der Funktionsbeschreibung auch ein Kommentar für jeden Parameter und den Rückgabewert angegeben. Das sieht dann in etwa folgendermaßen aus:

```
/**
 * Diese Methode hat keine besondere Funktionaliaet. Sie nimmt
 * zwar 2 Parameter entgegen und liefert einen davon zurück,
 * manipuliert diesen jedoch nicht.
 *
 * @param a Der erste Parameter.
 * @param b Der zweite Parameter.
 * @return Der Rueckgabewert.
 */


public int myMethod(int a, int b)
{
  return a;
}
```

Neben den Kommentaren, die von JAVADOC ausgewertet werden, finden komplizierte Stellen im Quelltext durch weitere Kommentare Erläuterung. Diese werden ausschließlich durch // eingeleitet und gelten bis zum Ende der Zeile. Um mehrzeilige Kommentare zu erzeugen, muß // jeder dieser Zeilen vorangestellt werden, wie das folgende Beispiel zeigt.

```
...
// Hier wird die Geschwindigkeit mit der Zeit multipliziert,
// um so die zurueckgelegte Strecke zu erhalten.
int distance = speed * time;
...
```

9.2 Bildverarbeitung mit IMAGEJ

Wie bereits in Kapitel 4.2 und eingangs des Kapitels 9.2 erwähnt, wurde das Bildverarbeitungsprogramm IMAGEJ als Grundlage für die Implementierung gewählt. IMAGEJ kann leicht um eigene Operatoren erweitert werden, die dann sämtliche Klassen und Operatoren von IMAGEJ benutzen können. In IMAGEJ existieren bereits Import- und Exportfunktionen für gängige Grafikdateiformate wie *.jpg*, *.gif*, und *.tif*. Insbesondere war schon eine Importfunktion für DICOM-Daten von Dr. Hackländer implementiert worden. Diese basierte zunächst auf einer fremden JAVA-Bibliothek und wurde später durch eine Minimalimplementierung des DICOM-Standards ersetzt [DICOM98], [SCHA99]. Auf den Import der Bilddaten wird in Kapitel 9.4 noch weiter eingegangen.

9.2.1 *Plugin* und *PluginFilter*

Die Schnittstellen *Plugin* und *PluginFilter* bieten die Möglichkeit, das Programm IMAGEJ um eigene Operatoren zu erweitern.

Um die Schnittstelle *Plugin* zu implementieren, muß lediglich die Methode *run* vorhanden sein. Bei dem *Plugin* zur Bildskalierung ergibt sich z.B. folgender Code:

```
public class BicubicImageScaler implements PlugIn
{
  ...
  public void run (String arg)
  ...
}
```

Das ist die Methode, die von IMAGEJ aufgerufen wird, wenn der Menüeintrag ausgewählt wird. Die Eintragung in die Menüstruktur erfolgt über die Datei *ij.properties*. Hier sind unter anderem die *Plugins* definiert:

```
plug-in08="DICOM import",DICOM_import
plug-in09="RENAL FUNCTION Analysis",RENAL_FUNCTION_Analysis
plug-in10="Bicubic Scaler",Bicubic_Scaler
```

Mit diesen Zeilen werden der DICOM-Import, die renale Funktionsanalyse und die bikubische Bildskalierung als *Plugin* konfiguriert. Optional können auch Parameter angegeben werden, die IMAGEJ der Methode *run* in *arg* übergibt. In obigem Fall ist *arg* leer.

Für die Schnittstelle *PluginFilter* muß noch eine weitere Methode namens *setup* implementiert werden. Diese wird einmal aufgerufen, danach wird *run* für jede Schicht eines Bilderstapels aufgerufen. Die renale Funktionsanalyse muß auf alle Schichten **gleichzeitig** zugreifen können, daher wurde sie als *Plugin* implementiert.

9.2.2 *ImagePlus* und *ImageStack*

Die Klasse *ImagePlus* ist eine erweiterte Klasse für Bilder, die neben Bildern mit 8-Bit Pixeldaten auch Bilder mit 16-Bit vorzeichenbehafteten Pixeldaten, 32-Bit Fließkomma und 32-Bit RGB unterstützt. *ImageStack* ermöglicht den Zugriff auf einen ganzen Stapel von Bildern. Dieser Bilderstapel (engl. stack) kann dynamisch verändert oder erweitert werden.

Die untersuchten Nierensequenzen liegen als 16-Bit vorzeichenbehaftete Grauwertbilder vor. Aufgrund der Turbo-Flash-Aufnahmetechnik werden davon jedoch nur die untersten 8 Bit genutzt, in den meisten Fällen sogar nur 7 Bit. Zur Reduzierung des Speicherbedarfs arbeitet die Kepstrum-Filterung daher auf 8-Bit vorzeichenbehafteten Grauwertbildern. Der Importfilter für DICOM-Dateien erzeugt stets eine Instanz von *ImageStack* mit der im DICOM-Format vorgegebenen Bit-Tiefe, in der die eigentlichen Pixeldaten gespeichert werden.

ImagePlus bietet die Möglichkeit, Attribute als Eigenschaften (engl. properties) in einer Instanz der Klasse *Properties* zu speichern. Die einzelnen Attribute werden dabei in einer *Hashtable* gespeichert. Auch der DICOM-Importfilter benutzt diese Tabelle, um dort die DICOM-Informationen abzulegen. Die Klassen zur renalen Funktionsanalyse greifen daher über

```
Properties prop = imp.getProperties();
String[] sl = null;

if (prop != null)
{
  sl = (String[]) prop.get("DICOM");
} else {
  // Keine Dicom-Informationen gefunden, Berechnung abbrechen.
  return;
}
```

auf die DICOM-Informationen zu. Die DICOM-Daten liegen jetzt als Feld vom Typ *String* vor und können auf bestimmte Einträge durchsucht werden. Sollte der Verweis auf die Properties leer sein, kann eine Verarbeitung der Sequenz dennoch durchgeführt werden, siehe Kapitel 10.2.3.

In *ImageStack* ist ein Attribut *stack* implementiert, das die Pixeldaten als Feld von Objekten vom Typ *Object* enthält. Die allgemeine Definition als *Object[]* ermöglicht die Speicherung der obengenannten verschiedenen Bildtypen. Der Zugriff auf die Pixeldaten wird mit

```
ImagePlus  imp = WindowManager.getCurrentImage();
ImageStack is  = imp.getStack();
```

erreicht. Auf die einzelnen Schichten im Stapel, entsprechend den Bildnummern der Nierensequenz, kann dann über die Methode *getPixels* in der Klasse *ImageStack* zugegriffen werden:

```
private short[] filterSlice(int nr)
{
  short pix[] = new short[imp.getWidth()*img.getHeight()];
  pix = (short[]) is.getPixels(nr);
  // Nun koennen die Pixel der Schicht manipuliert werden:
  // doManipulations(pix);
  return pix;
}
```

Dabei ist *nr* die Nummer der Schicht. Die modifizierten Pixeldaten können entweder in einem neuen Stack abgelegt werden:

```
newStack.addSlice("neue Daten", (Object) pix);
```

oder über die des alten Stacks geschrieben werden:

```
is.setPixels((Object) pix, nr);
```

9.2.3 *ImageWindow* und *ImageCanvas*

Die Klasse *ImageWindow* realisiert die Anzeige der in *ImagePlus*-Objekten gespeicherten Bilder. IMAGEJ geht dabei anders vor als übliche Grafikprogramme, denn jedes Bild erhält sein eigenes Fenster und wird nicht als Unterfenster innerhalb eines Programmhauptfensters dargestellt. Die

Ursache hierfür liegt im JAVA-Grafikpaket AWT[12], das IMAGEJ ausschließlich implementiert. Dort fehlen die Klassen zum Erstellen einer Mainframe-Child-GUI-Architektur[13].

ImagePlus und *ImageWindow* verweisen mit gegenseitigen Referenzen aufeinander. Das *ImageWindow* wird erst dann initialisiert, wenn die Methode *show* von *ImagePlus* aufgerufen wird. In einer Statuszeile über dem Bild zeigt das *ImageWindow* zusätzliche Informationen über die Bildeigenschaften an, siehe Abbildung 9.1.

Besteht das *ImagePlus*-Objekt verwaltete Bild aus einem Stack, so wird zur Anzeige die von *ImageWindow* abgeleitete Klasse *StackWindow* genutzt. Diese erweitert *ImageWindow* im wesentlichen nur um einen Schieberegler unter dem Bild, mit dem die aktuell angezeigte Schicht des Stacks einstellbar ist.

```
ImagePlus  imp = new ImagePlus(„New Image“,(Object)pixel);
imp.show();
```

Die Klasse *ImageCanvas* realisiert die eigentliche Anzeige der Pixeldaten sowie der eingezeichneten ROI. Sie ist Bestandteil eines jeden *ImageWindow*-Objektes. Sie implementiert Low-Level-Routinen für das Mouse-Handling sowie Zoom-Funktionalitäten. Um mehrere ROIs in einem Bild einzeichnen zu können, wird sie später noch abgeleitet und stark erweitert, siehe Kapitel 9.8.2.

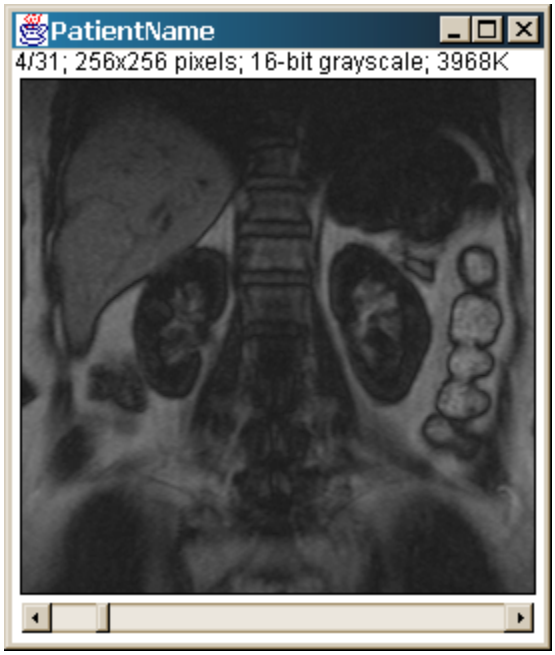


Abbildung 9.1: Ein StackWindow-Objekt mit Schieberegler. Der ImageCanvas stellt das Bild dar.

9.3 Übersicht

In Abbildung 9.2 ist nochmals das bereits in Kapitel 4.7 gezeigte Diagramm der Systemkomponenten und des Bearbeitungsweges dargestellt. Alle im Rahmen dieser Arbeit entstandenen Teile sind dabei grau hinterlegt, ihre Struktur und Implementierung werden im folgenden behandelt. Während der Implementierung wurden verschiedene JAVA-Pakete[14] erstellt, die sich durch die Zusammenfassung jeweils der Klassen auszeichnen, die eine funktionelle Einheit bilden. Lediglich die Klassen der

[12] Die Grafikbibliothek AWT (kurz für engl. abstact windowing toolkit) ist Bestandteil des JDK seit der Version 1.0
[13] Mainframe ist die englisch Bezeichnung für Hauptfenster, Childframe steht für Kindfenster.
GUI steht kurz für graphical user interface, zu deutsch grafische Benutzeroberfläche.
[14] JAVA-Pakete (engl. packages) stellen eine Möglichkeit zur Verfügung, zusammengehörende oder in Beziehung stehende Klassen und Schnittstellen zu einer Einheit zusammenzufassen.

Benutzerschnittstelle sind nicht zu einem Paket zusammengefaßt, sondern ordnen sich denjenigen Paketen unter, deren Berechnungsergebnisse sie darstellen. In Abbildung 9.2 sind den Programmkomponenten die jeweiligen Pakete in kursiver Schrift zugeordnet.

Ein allgemeines Problem der Paket- und Klassenstruktur besteht darin, daß das System aus dem Zusammenspiel verschiedener Teillösungen entstanden ist. Deren Umfang und Struktur hat sich meistens nach den Ergebnissen einer programmiererischen Testphase und der benötigten Funktionalität gerichtet. Die anfangs gedachte Programmstruktur wurde mehrmals geändert und korrigiert.

Im folgenden wird nun die Implementierung anhand der funktionellen Programmteile erläutert, die Beschreibung der in jedem Programmteil benutzten Pakete ordnet sich den Kapiteln unter.

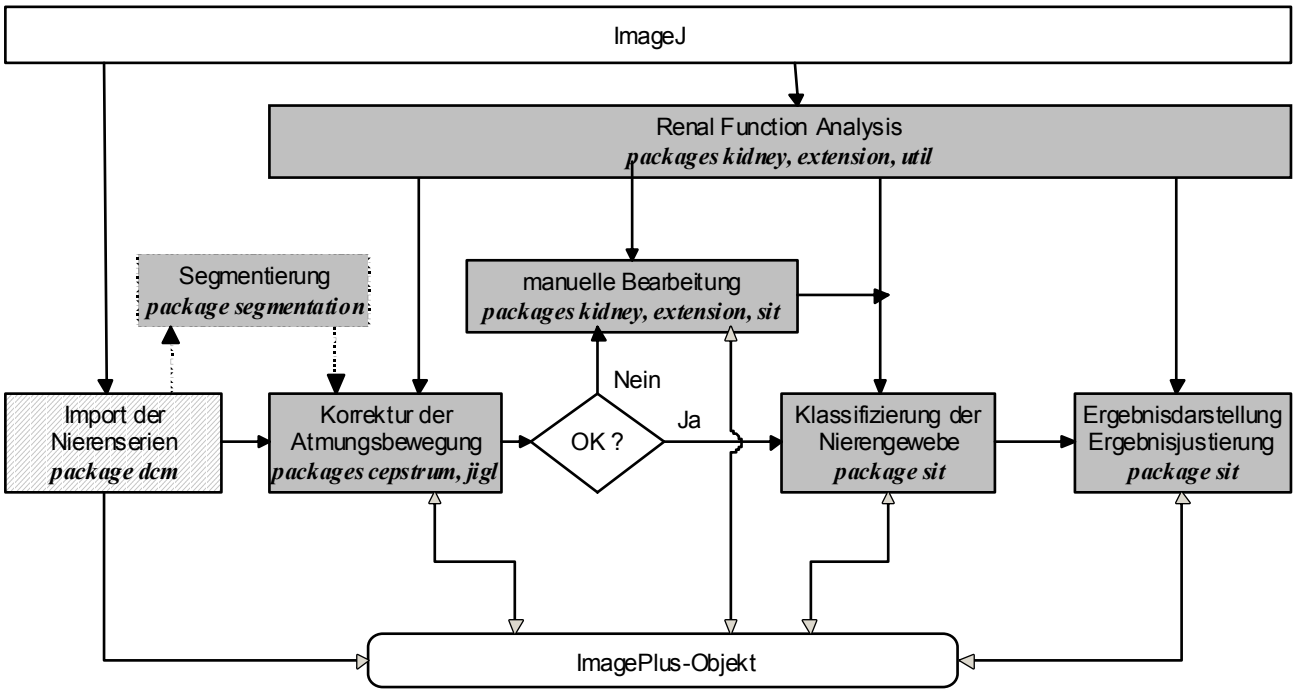


Abbildung 9.2: Zuordnung der Pakete zu den Programmkomponenten. Alle grau hinterlegten Teile sind im Rahmen dieser Arbeit entstanden.

9.4 Import der Nierensequenzen

Wie bereits in Kapitel 4 erwähnt, ist der Import der Nierensequenzen als *Plugin* für IMAGEJ implementiert. Die dazu nötigen Klassen wurden im JAVA-Paket *dcm* zusammengefaßt. Das *Plugin* wurde bereits vor dem Zeitpunkt der Implementierung dieser Arbeit entwickelt und konnte daher für den Import der Nierensequenzen eingesetzt werden. Das *dcm*-Paket befindet sich jedoch in einem ständigen Prozeß der Weiterentwicklung unter der Beteiligung von Dr. Hackländer sowie einigen ehemaligen Mitgliedern der Projektgruppe 305 der Universität Dortmund [RFH98]. Die Mitarbeit des Autors im Rahmen dieser Arbeit beschränkte sich auf einige Anpassungen sowie Hilfsklassen, daher sollen hier lediglich die verwendeten Import-Klassen diskutiert werden. Die Struktur des *dcm*-Paketes soll aber in ihren Einzelheiten hier nicht ausgeführt werden.

Die Klasse *DICOM_import* erweitert die IMAGEJ-Klasse *ImagePlus* und implementiert die Schnittstelle *Plugin*. Sie steuert den Aufruf des Benutzerdialoges zur Auswahl der Serie, die Klasse *PanelDialog*. Diese liefert eine Liste der ausgewählten Dateien an *DICOM_import* zurück, die dann unter Benutzung der im *dcm*-Paket implementierten Low-Level-Klassen der DICOM-Schnittstelle den

Import vornimmt [DICOM98]. Die Bildsequenz wird in einen *ImageStack* umgewandelt, wobei die zusätzlichen Aufnahme- und Patienteninformationen in einer Instanz eines *Properties*-Objektes gespeichert werden. Auf das *ImageStack*-Objekt existiert im *ImagePlus*-Objekt eine Referenz. Letztendlich wird durch den Aufruf der Methode *show* ein *ImageWindow*- oder *StackWindow*-Objekt instanziiert und dargestellt. Die folgende Abbildung 9.3 zeigt die Zusammenhänge der gerade besprochenen Klassen. Da es zu unübersichtlich werden würde, alle Klassen des *dcm*-Paketes darzustellen, wurden diese weitgehend weggelassen.

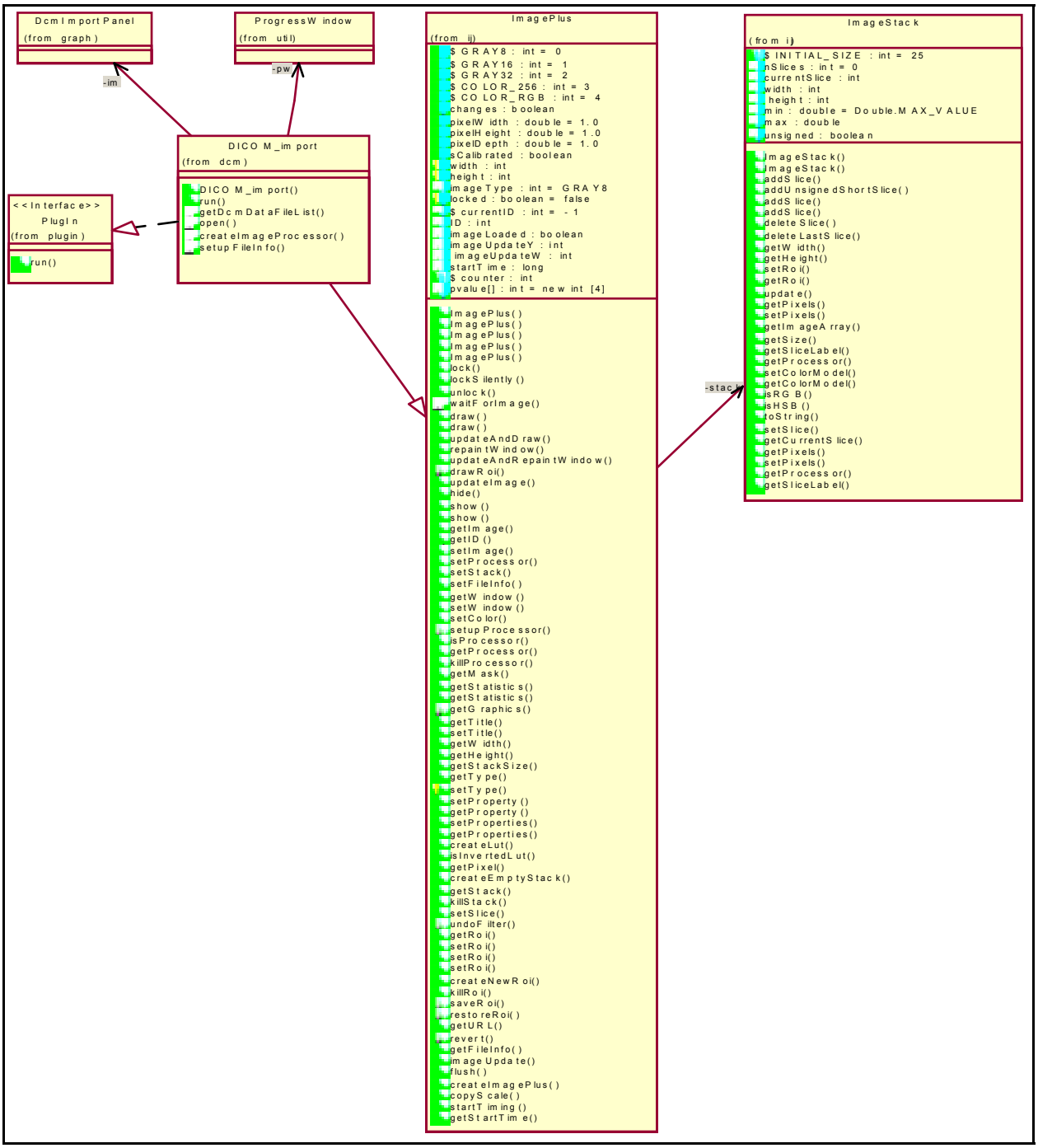


Abbildung 9.3: Klassendiagramm der zum Import benötigten Klassen.

9.5 Das *Plugin* zur renalen Funktionsanalyse

Das *Plugin* zur renalen Funktionsanalyse wird von der Klasse *RENAL_FUNCTION_Analysis* implementiert. Diese prüft zunächst, ob überhaupt eine Sequenz importiert wurde und ob diese evtl. gerade von einem anderen *Plugin* oder *Filter* bearbeitet wird. Weiter wird getestet, ob die importierte Sequenz vom richtigen Format ist. Die renale Funktionsanalyse kann nur auf Grauwertbildern mit 8-, 16- oder 32-Bit Tiefe durchgeführt werden. Letztendlich muß noch die Anzahl der Bilder in der importierten Sequenz geprüft werden, da die Implementierung im weiteren eine Anzahl von mindestens 2 Bildern voraussetzt und dies nicht weiter geprüft wird. Sinnvolle Ergebnisse lassen sich erst ab einer Serie von 15-20 Bildern berechnen. Wenn alle Tests erfolgreich waren, wird schließlich der Dialog des *Plugins* instanziiert und angezeigt.

```
ImagePlus imp;
imp = WindowManager.getCurrentImage();
if (imp!=null)
{
  if (!imp.lock())
  {
    return;   // exit if image is in use
  }
  win = imp.getWindow();
}
// If there is a Window, test the Type of Image
// Supported: GRAY 8,16,32   Not Supported: the Rest
int imageType = imp.getType();
if (imageType==imp.COLOR_RGB || imageType==imp.COLOR_256)
{
  IJ.error("Kidney Analysis is limited to \n8bit Gray, 16bit
    Gray, or 32bit Gray Dicom-Images !");
  return;
}
```

9.5.1 Das Paket *kidney*

Das Paket *kidney* war das erste erstellte Paket und enthält den Hauptdialog des *Plugin*, die Klasse *AnalysisDialog*. Weiter sind noch die Klassen *PresetsDialog* zur Realisierung des Dialoges mit Voreinstellungen sowie die später noch behandelte Klasse *ManualMover* enthalten, siehe Kapitel 9.8.1. Die Klasse *AnalysisDialog* bekommt vom aufrufenden *Plugin* eine Referenz auf das *ImagePlus*-Objekt übergeben, das die Nierensequenz enthält. Durch den Aufruf der Methode *getProperties* werden zunächst die Eigenschaften der Nierensequenz ausgelesen und in Variablen gespeichert. Dies sind im wesentlichen Patienteninformationen, die in *Strings* abgelegt werden (Name, Geschlecht, Alter, Gewicht) sowie Aufnahmeparameter (Auflösung, Pixelfläche, Anzahl der Bilder, Zeitpunkt der Aufnahme). Für die weitere Analyse ist insbesondere der zeitliche Abstand der Bilder von Interesse, da dieser für die Berechnung der SIT-Kurven wichtig ist. Dem ersten Bild der Sequenz wird der Aufnahmezeitpunkt 0 zugeordnet, für alle weiteren Bilder kann der Abstand in Millisekunden anhand der Properties berechnet werden und wird in einem *int[]* Array abgelegt.

Wie bereits in Kapitel 9.2.3 erwähnt, mußte der in IMAGEJ implementierte *ImageCanvas* in seiner Funktionalität erweitert werden. Das Einzeichnen mehrerer ROIs sowie die Darstellung zusätzlicher Informationen und Schaltflächen (engl. buttons) war nötig, um eine bedienbare Benutzeroberfläche zur renalen Funktionsanalyse zu erstellen. Daher wurden die IMAGEJ-Klassen *ImageCanvas*, *ImageWindow* und *StackWindow* abgeleitet und erweitert. Eine Beschreibung der daraus entstandenen Klassen *ExImageCanvas*, *ExImageWindow* und *ExImageStack* wird in Kapitel 9.5.2 gegeben. Die Klasse *AnalysisDialog* enthält jeweils eine Referenz auf ein *ExImageCanvas* und *ExStackWindow*-

Objekt. Das dem Konstruktor der Klasse übergebene *ImagePlus*-Objekt enthält eine Referenz auf das darstellende *ImageWindow*, dieses enthält wiederum eine Referenz auf den *ImageCanvas*. Durch explizite Typumwandlung werden nun die Referenzen in *AnaylsisDialog* gesetzt:

```
ExStackWindow exStackWindow;
ExImageCanvas exImageCanvas;
exStackWindow = new ExStackWindow(imagePlus);
exImageCanvas = (ExImageCanvas)exStackWindow.getCanvas();
```

Die vollzogene Typumwandlung ist vom Benutzer sofort wahrnehmbar durch die zusätzlichen Textfelder und Buttons auf dem *ImageCanvas* sowie den schwarzen Hintergrund. Alle GUI-Komponenten der renalen Funktionsanalyse passen sich damit den bei MRT-Arbeitsplätzen häufig eingesetzten Farbschemata an, Bilder und Text in heller (weißer) Schrift auf dunklem (schwarzem) Hintergrund darzustellen. Dies entspricht zwar nicht den Richtlinien zur Oberflächengestaltung [HORN98], [GEIS98], jedoch wird einer Empfehlung von FELLNER Folge geleistet, Bildmaterial mit schwarzem Hintergrund in einer Umgebung mit dunklem Hintergrund darzustellen. Dadurch wird die Helligkeits- und Kontrastwahrnehmung des menschlichen Auges weniger beeinflußt [FELL90].

Ursprünglich war die Klasse *AnalysisDialog* lediglich zur Benutzerinteraktion gedacht. Aufgrund eines fehlenden globalen Datenobjektes, in dem z.B. die Bilddaten, die Properties sowie sämtliche berechnete Zwischen- und Endergebnisse gespeichert werden könnten, wurde auch diese Funktion durch den *AnalysisDialog* übernommen. Zur Laufzeit existiert genau eine Instanz dieser Klasse, deren Referenz an die verschiedenen Klassen zur Berechnung der Kepstrum-Filterung sowie der SIT-Kurven übergeben wird. Einige Variablen der Klasse, z.B. die Voreinstellungen, sind daher für den einfacheren Zugriff als *public* deklariert.

Der von der Klasse *AnalysisDialog* implementierte Benutzerdialog enthält im wesentlichen textuelle Informationen über die Nierensequenz sowie Schaltflächen zum Aufruf der einzelnen Berechnungskomponenten des Systems. Der Aufruf der Komponenten erfolgt dabei interaktiv, da i.d.R. vor Beginn der Kepstrum-Filterung bzw. der Berechnung der SIT-Kurven ROIs einzuzeichnen oder anzupassen sind. Die Aufrufe der Berechnungsklassen erfolgen über die Methoden *startCepstrum*, *startSIT* und *startSegmentation*. Diese instanziieren jeweils Objekte der Berechnungsklassen und starten deren *Threads*. Zur Vermeidung von Fehleingaben während eines Berechnungszyklus werden alle Komponenten des Dialogs sowie das *ExStackWindow* für Eingaben gesperrt:

```
this.enableButtons(false);
exStackWindow.lock();
```

Sind die jeweiligen Berechnungen beendet, so rufen diese die Methoden *cepstrumFinished*, *sitFinished* und *segmentationFinished* auf und aktivieren die Benutzerschnittstelle damit wieder.

Die Klasse *PresetsDialog* implementiert die Eingabe und die Auswahl von Voreinstellungen, die für die Berechnungen notwendig sind. Die Parameter sind dabei entsprechend ihrer Zuordnung zur Kepstrum-Filterung oder SIT-Berechnung unterteilt und werden durch zwei *JTabbedPanes* dargestellt. Ein drittes *Tab* für globale Einstellungen steuert die Priorität der *Threads*.

Bei der Instanziierung des Benutzerdialoges der Klasse *AnalysisDialog* wird eine Instanz von *PresetsDialog* erzeugt und referenziert. Die Anzeige des Voreinstellungs-Dialoges wird jedoch erst dann durchgeführt, wenn der *Presets-Button* im Hauptdialog betätigt wird. Wird das Voreinstellungsfenster wieder geschlossen, so wird die Instanz der *PresetsDialog*-Klasse nicht gelöscht, sondern bleibt durch eine Referenz in *AnalysisDialog* erhalten. Die Methode *loadPresetsFile* wird vom Konstruktor von *PresetsDialog* aufgerufen und lädt die zuletzt gespeicherten Voreinstellungen aus einer Datei mit dem Namen *RenalFunctionAnalysis.properties*. Dort sind die Voreinstellungen in Textform gespeichert, was den Vorteil hat, daß die Datei in Ausnahmefällen mit auch jedem Texteditor erstellt oder verändert werden kann. Bei jedem Verlassen des Voreinstellungs-Dialoges über den *Exit-Button*

werden veränderte Parameter in der Datei aktualisiert. Dies geschieht über den Aufruf der Methode *savePresetsFile*.

Die auch in diesem Paket zugeordnete Klasse *ManualMover* wird in Kapitel 9.8.1 im Zusammenhang mit der manuellen Bearbeitung beschrieben.

9.5.2 Das Paket *extension*

Dieses Paket enthält alle erweiterten Klassen, d.h. alle von anderen Klassen abgeleiteten und funktionell ergänzten Klassen. Wie bereits erwähnt, war dies besonders im Fall der Klasse *ImageCanvas* nötig. Die Klasse *ExImageCanvas* überschreibt die Methode *paint* ihrer Oberklasse und kann so zusätzliche Informationen, Buttons, Tooltips[15] sowie mehrere ROIs auf dem *Canvas* einzeichnen. In *paint* wird daher zunächst die *paint*-Methode der Oberklasse aufgerufen, die das eigentliche Bild einzeichnet. Danach werden die zusätzlich angezeigten Objekte auf dem *Canvas* eingezeichnet und befinden sich so „über“ dem Bild. Für die Darstellung dieser zusätzlichen Informationen werden nur im obersten und untersten Bildbereich ein ca. 10 Pixel breiter Streifen benötigt. In allen vorliegenden Nierensequenzen werden dadurch weder die Nieren selbst noch Teile davon überdeckt.

Die Methode *setAcquisitionTime* nimmt von der im vorigen Kapitel beschriebenen Klasse *AnalysisDialog* die Aufnahmezeitpunkte der einzelnen Bilder der Sequenz entgegen. *AnalysisDialog* liefert dabei ein *string[]*-Array fertig formatierter *Strings*, die am unteren Rand des Bildes dargestellt werden.

Durch die Methode *setCepstrumResult* wird für ein bestimmtes Halbbild eines Sequenzbildes das Ergebnis der Kepstrum-Filterung gesetzt. Dieses wird ebenfalls als Textinformation über das Bild gelegt, am oberen Bildrand für jede Bildhälfte getrennt, siehe auch Abbildung 10.9. Ist das Ergebnis dabei besser als ungenügend (und damit der übergebene String nicht "––"), so markiert *setCepstrumResult* automatisch das angegebene Halbbild als gültig für die Berechnung der SIT-Kurven, siehe auch Diagramm in Abbildung 4.5.

Um die Möglichkeit zu bieten, mehrere ROIs einzeichnen zu können, wird in *ExImageCanvas* für jedes Sequenz-Halbbild ein Array zur Speicherung von bis zu 100 ROIs pro Gewebeart (Medulla und Cortex) bereitgestellt. Zusätzlich müssen pro Sequenzbild zwei Haupt-ROIs (entlang der Nierenkontur) eingezeichnet werden können; diese werden ebenfalls in einem Array verwaltet. Für das Setzen und Löschen der ROIs existieren mehrere Methoden. Es ist stets möglich, eine aktive ROI entweder nur für das aktuelle Bild festzulegen oder für alle Bilder der Sequenz einzutragen. Entsprechend des gewünschten Gewebes wird die aktive ROI mit der Methode *getRoi* im *ImagePlus* gelesen und in ein *Roi[]*-Array das *Canvas* eingetragen:

Indem *ExImageCanvas* einen *KeyListener* implementiert, kann abgefragt werden, ob der Benutzer die ALT-Taste oder die ENTF-Taste gedrückt hält, während er durch einen Klick auf einen der Buttons eine ROI festlegt. Damit kann die Bedienung etwas vereinfacht werden, siehe auch Kapitel 10.2.7. Alle übrigen Methoden von *ExImageCanvas* werden weitgehend für die Behandlung von Mausklicks und -bewegungen benötigt und sollten sich aus den Kommentaren im Quelltext erklären.

Die Klassen *ExImageWindow* und *ExStackWindow* entsprechen fast gänzlich ihren Oberklassen von IMAGEJ. Lediglich die enthaltenen Variablen für die (gegenseitigen) Verweise wurden in ihrem Typ verändert und erwarten nun abgeleitetes Objekt *ExImageCanvas*, *ExStackWindow* oder *ExImageWindow*. Diese Anpassungen waren nötig, um die in Kapitel 9.5.1 beschriebene explizite Typumwandlung des *ImageCanvas* in einen *ExImageCanvas* sauber durchführen zu können. In *ExStackWindow* wurden noch die beiden Funktionen *lock* und *unlock* hinzugefügt. Durch Ihren Aufruf kann der Schieberegler zum Durchfahren der Sequenz sowie der *Canvas* für die Dauer einer Berechnung deaktiviert bzw. wieder aktiviert werden.

[15] Beschreibungsfelder, die beim Positionieren der Maus über einem GUI-Element eingeblendet werden.

Zur Realisierung von Tooltips auf dem *Canvas* wurden noch die Hilfsklassen *AwtToolTip* und *AwtToolTipHelper* erstellt. Tooltips werden als Mittel zur Selbstbeschreibung von allen Komponenten der Benutzerschnittstelle genutzt; dadurch ist eine intuitivere Bedienung möglich. Außerdem helfen Tooltips oft dem Benutzer, sich nach längerer Pause schneller wieder in die Bedienung des Programms einzufinden, ohne im Handbuch nachschlagen zu müssen. Tooltips hielten in JAVA allerdings erst mit der SWING-Klassenbibliothek Einzug; das Standard-AWT-Toolkit (und somit auch IMAGEJ) kann keine Tooltips darstellen. Die Klassen *AwtToolTip* und *AwtToolTipHelper* ermöglichen dies nun auch für AWT-Objekte. Dabei werden die Einstellungen (Farbe, Verzögerungszeit, etc.) vom SWING-Toolkit übernommen:

```
AwtToolTip awtToolTip = new AwtToolTip();
AwtToolTipHelper awtToolTipHelper = new AwtToolTipHelper();
JToolTip toolTip = new JToolTip(); // Ein Swing-Tooltip erzeugen

awtToolTip.setDismissDelay(ToolTipManager.sharedInstance().getDi
smissDelay());
awtToolTip.setInitialDelay(ToolTipManager.sharedInstance().getIn
itialDelay());
toolTipLabel.setBackground(toolTip.getBackground());
toolTipLabel.setForeground(toolTip.getForeground());
toolTipLabel.setFont(toolTip.getFont());
```

Das Anzeigen der Tooltips wird in der *paint*-Methode von *ExImageCanvas* durchgeführt. Das referenzierte *AwtToolTip*-Objekt kann mit Hilfe der Methode *awtToolTip.showTooltip* befragt werden, ob eine Anzeige stattfinden soll.

9.5.3 Das Paket *util*

Das Paket *util* enthält die beiden Klassen *JMConst* und *JMUtil*. Ursprünglich war gedacht, häufig verwendete Methoden in der Klasse *JMUtil* bereitzustellen, um Klassen der anderen Pakete schlanker zu machen und doppelten Code zu vermeiden.

Die Klasse *JMConst* definiert für die Berechnung notwendige numerische oder textuelle Konstanten. Diese sind als *public final static* deklariert und somit von allen Methoden nicht veränderbar. Der Hintergedanke ist eine globale Definition der Konstanten, um eine mehrfache Neudefinition derselben Konstanten im Kopf verschiedener Klassen zu umgehen. Somit werden Fehler vermieden und der Quelltext lesbarer. Für den Zugriff auf die Konstanten muß lediglich *JMConst* importiert werden, mit

```
import jm.util.JMConst;
int imageHalf = JMConst.LEFT;
int tissue = JMConst.CORTEX;
```

können die Werte dann ausgelesen werden.

Die folgende Abbildung 9.4 gibt wieder einen zusammenfassenden Überblick der gerade besprochenen Klassen.

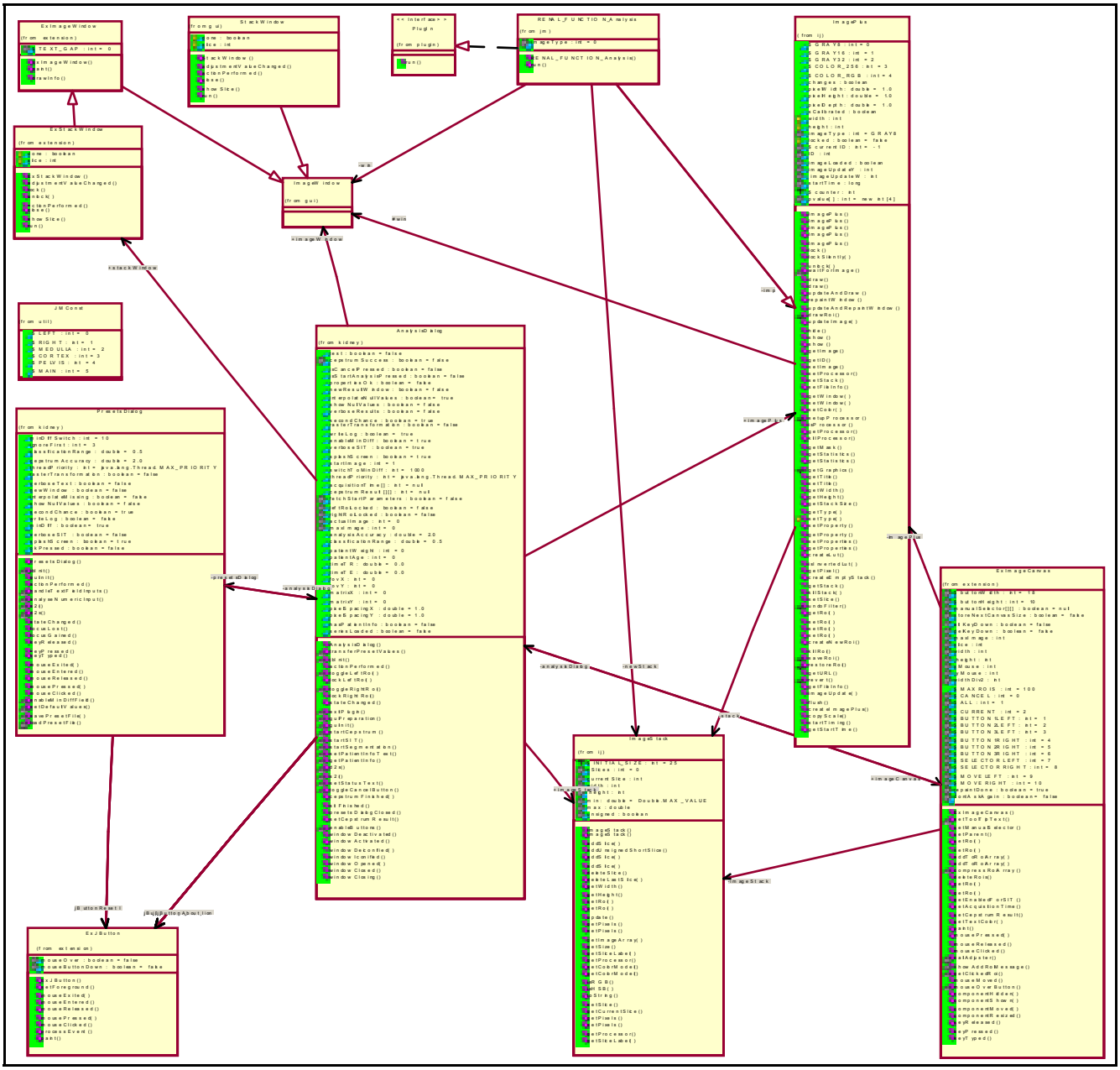


Abbildung 9.4: Übersicht der für die Realisierung des *Plugins* zur renalen Funktionsanalyse benötigten Klassen.

9.6 Kepstrum-Filterung

Alle zur Durchführung der Kepstrum-Filterung notwendigen Klassen sind im Paket *cepstrum* zusammengefaßt. Die Fourier-Transformation als elementarer Bestandteil der Kepstrum-Filterung (vergleiche Kapitel 5.3.1 und 6.2) wurde jedoch nicht neu implementiert, sondern unter Verwendung der frei verfügbaren Bibliothek *JavaImageGraphicsLibrary* der BRIGHAM YOUNG UNIVERSITY ins Programm eingebracht.

9.6.1 Das Paket *jigl*

Die Bezeichnung *jigl* steht als Kürzel für *JavaImageGraphicsLibrary*. Von dieser sehr umfangreichen Bibliothek wurden lediglich diejenigen Klassen übernommen, die zur Durchführung der Fourier-Transformation bzw. zur Bildnachverarbeitung und Bilddarstellung benötigt werden. Die Darstellung des bei der Fourier-Transformation berechneten Bildes im komplexen Raum, zusammengesetzt aus Real- und Imaginärteil, kann dabei durch Verwendung des *jigl*-Datentyps *ComplexImage* erreicht

werden. Der in der Bibliothek enthaltene Fourier-Algorithmus ist in der Klasse *FFT*[16] implementiert und erwartet als Eingabe ein Bild vom Typ *RealGrayImage*. Die importierten DICOM-Bilder liegen, wie bereits erwähnt, aber als *ImagePlus*-Objekt vor. Da einerseits die *jigl*-Klasse *RealGrayImage* einen Konstruktor mit Eingabe eines *java.awt.Image* besitzt, andererseits *ImagePlus* eine Methode implementiert, die ein *java.awt.Image* liefert, ist dies der eleganteste Weg:

```
// Konvertieren eines ImagePlus-Bildes in ein jigl.RealGrayImage
java.awt.Image transferImage =
  imagePlus.getProcessor().createImage();
 jigl.RealGrayImage realGrayImg =
  new jigl.RealGrayImage(transferImage);

// Durchfuehren der FFT
jigl.ComplexImage complexImg = jigl.FFT.doFFT(realGrayImg,true);
```

Der *boolesche* Wert als zweiter Parameter bestimmt, ob eine *FFT* oder eine inverse *FFT* durchgeführt wird: Ist der Parameter **true,** wird eine Fourier-Transformation durchgeführt, ist er dagegen **false**, wird eine inverse Fourier-Transformation durchgeführt. Das fourier-transformierte Bild kann also mit

```
refftComImg = jigl.image.utils.FFT.doFFT(finalFFTImg,false)
```

wieder zurücktransformiert werden. Da die Berechnung des Kepstrum-Filters aber ausschließlich Vorwärts-Transformationen einschließt, wird dieser Fall nicht benötigt. Zuletzt wird aus dem Ergebnis ein Magnitudenbild berechnet, siehe auch Kapitel 5.3.1:

```
for (int x=0; x<width; x++)
{
  for (int y=0; y<height; y++)
  {
    MagnitudeImage[x][y]=(int)
      (Math.sqrt(Math.pow(complexImg.getReal(x,y),2) +
                 Math.pow(complexImg.getImag(x,y),2) ) );
  }
}
```

Die *FFT*-Methode des *jigl*-Paketes war zwar sauber implementiert, konnte jedoch noch weiter optimiert werden. Es wurden dabei einige Feinabstimmungen durchgeführt, um die Geschwindigkeit zu steigern. Meist gingen diese zu Lasten des benötigten Speicherbedarfs, aber bei einer maximalen Bildgröße von 512×512 Pixeln als Eingabebild fiel dies kaum ins Gewicht. Außerdem wird die Referenz auf das *FFT*-Klassenobjekt gelöscht, sobald dieses nicht mehr benötigt wird. Für die am häufigsten vorkommende Eingabegröße eines 256×256 Pixel großen Kepstrum-Fensters konnte die Rechengeschwindigkeit um ca. 45% von 7,0 auf durchschnittlich 3,7 Sekunden gesteigert werden[17].

9.6.2 Das Paket *cepstrum*

Dieses Paket umfaßt die vier Klassen *ResultLog*, *BicubicInterpolation*, *Cepstrum* und *CepstrumAnalysis*. Während die Klasse *CepstrumAnalysis* den Bearbeitungsablauf der Nierensequenz steuert (vergleiche Abbildung 4.5), wird in *Cepstrum* die eigentliche Berechnung der Kepstrum-Filterung durchgeführt.

CepstrumAnalysis wird als Objekt von *AnalysisDialog* (siehe Kapitel 9.5.1) instanziiert und bekommt im Konstruktor eine Referenz auf das aufrufende *AnalysisDialog*-Objekt sowie die ausgewählte (rechteckige) ROI um die linke und rechte Niere übergeben. Anhand der eingestellten Berechnungsgenauigkeit wird zunächst festgelegt, ob und auf welche Größe die beiden Eingabebilder

[16] FFT steht für FastFourierTransform, siehe auch Kapitel 7.3

[17] Die Angaben beziehen sich auf einen PENTIUM III 450 MHz Prozessor unter Verwendung des SUN JDK 1.1.8

für die Kepstrum-Filterung hochskaliert werden müssen. Nun kann die Abarbeitung der Nierensequenz beginnen.

Da *CepstrumAnalysis* direkt von *java.lang.Thread* abgeleitet ist, kann der Berechnungs-Thread durch den Aufruf der Methode *start* der Oberklasse erzeugt und gestartet werden. Die *Virtual Machine* ruft dann automatisch die Methode *run* in *CepstrumAnalysis* auf. Diese enthält im wesentlichen eine große *while*-Hauptschleife sowie mehrere *if*-Abfragen zur Bedingungsprüfung während der Berechnung. Da der Magnetresonanztomograph eine Einschwingphase von ein bis zwei Bildern benötigt, wird die Filterung der Sequenz i.d.R. nicht beim ersten Bild begonnen, sondern erst bei Bild 4 (per Voreinstellung festlegbar). Bei jedem Schleifendurchlauf wird zunächst der Bildinhalt innerhalb der gegebenen ROI für Bild *n* und *n*+1 ausgeschnitten und evtl. hochskaliert. Die Methode *prepareImages* führt dies durch. Dann findet entsprechend Kapitel 7.4 ein Angleich der Histogramme statt (Methode *histogramEquilisation*), bevor die Bilder in die Kepstrum-Filterung eingegeben werden. Alle weiteren Vorverarbeitungsschritte (vergleiche Kapitel 7.1, 7.2 und 7.3) sowie die Anordnung im Kepstrum-Fenster werden von der im folgenden Abschnitt diskutierten Klasse *Cepstrum* durchgeführt. Die in Kapitel 7.7.1 besprochene Nachjustierung wird von der Methode *checkCepstrumResult* berechnet. Die Korrektur des Folgebildes *n*+1 um die berechnete Verschiebung geschieht in *applyShift*, Dabei wird ein neuer *ImageStack* aufgebaut, der entweder den alten *ImageStack* ersetzt oder in einem neuen Fenster angezeigt werden kann. Alle weiteren (Hilfs-)Methoden der Klasse sind weitgehend durch ihre selbstbeschreibenden Namen sowie Kommentare im Quelltext erklärt.

Es sei kurz erwähnt, daß die *while*-Schleife für jedes Bild der Sequenz zweimal durchlaufen wird, da rechte und linke Bildhälfte unabhängig voneinander bearbeitet werden. Die Progammierung ist so ausgelegt, daß lediglich eine Variable *half* auf die Konstante LEFT oder RIGHT gesetzt werden muß, um die Auswahl der Bildhälfte zu steuern. Eine Auswertung in getrennten Threads wurde kritisch geprüft. Der Quelltext ist durch geringe Änderungen an eine solche Multi-Thread-Berechnung anzupassen. Das SUN JDK 1.1.8 führt Threads jedoch nur unter SOLARIS oder LINUX auf mehreren Prozessoren aus, wenn diese vorhanden sind. Unter WINDOWS9X und WINDOWS NT sowie auf dem MACOS werden alle Threads demselben Prozessor zugeordnet, auf dem auch die *Virtual Machine* läuft. Somit entsteht hier eher ein Geschwindigkeitsverlust, da durch die Thread-Verwaltung stets ein zusätzlicher Rechenaufwand entsteht.

Wie bereits gesagt, bekommt die Klasse *Cepstrum* das Referenz- und Folgebild als *java.awt.Image* beim Aufruf der Methode *calculateTranslation* übergeben. Der als dritter Parameter übergebene *java.awt.Point* enthält das Ergebnis der Verschiebungsberechnung. Die Methoden *filterSobel* und *applyWindow* führen die in Kapitel 7.1 und 7.2 besprochene Vorverarbeitung durch. Dann erst werden die Bilder im Kepstrum-Fenster angeordnet und die Berechnung durch den Aufruf der Methode *doCepstrum* durchgeführt. Die beiden als *public* definierten Variablen *peakQuality* und *peakAboveAverage* werden von der Klasse *CepstrumAnalysis* benötigt, um die Laufzeit-Ergebniskontrolle durchzuführen. Sie werden während der Detektion des Peaks im Suchbereich berechnet. Die Methode *findLocalMax* führt die Detektion aus.

Die Klasse *BicubicInterpolation* wird sowohl zum Hochskalieren von Referenz- und Folgebild eingesetzt sowie zur Verschiebung eines (Halb-)Bildes um nicht ganzzahlige Verschiebungswerte. Sie implementiert das in Kapitel 5.4.2 behandelte Verfahren zur bikubischen Interpolation. Da in der Regel stets mehrere interpolierte Pixel ein- und desselben Bildes berechnet werden, bekommt *BicubicInterpolation* im Konstruktor ein Array, das die Pixel des zu interpolierenden Bildes enthält, sowie die Höhe und Breite des Bildes übergeben. Durch Überladung[18] des Konstruktors können verschiedene Typen (*byte*, *short* und *int*) von Arrays übergeben werden und so 8-, 16- und 32-Bit Grauwertbilder interpoliert werden. Ist das Objekt instanziiert, kann mit der Methode *getPixel* der

[18] Die sogenannte Überladung ist ein programmiererisches Mittel, um eine Methode mit unterschiedlich typisierten Parametern aufrufen zu können.

Grauwert einer beliebigen, reellwertigen Bildposition ($x,y \in \mathbb{R}$) abgefragt werden. Die Schleife zum Verschieben eines Halbbildes um die Werte 11,3 / 4,8 sieht dann so aus:

```
BicubicInterpolation bic = new
  BicubicInterpolation(pix,width,height);
short[] interpolatedImage = new short[width*height];

for (int y=0; y<height; y++)
{
  for (int x=0; x<width/2; x++)
  {
    int gray = bic.getPixel((double)x – 11.3, (double)y – 4.8;
    interpolatedImage[(y*width)+x] = (short)val;
  }
}
```

Eine Hochskalierung des Bildes kann durch das Auslesen mehrerer Pixel innerhalb eines Raster-Intervalls durchgeführt werden. Die Skalierung auf 2,5-fache Größe wird beispielsweise berechnet mit:

```
BicubicInterpolation bic = new
  BicubicInterpolation(pix,width,height);
short[] scaledImage = new short[width*2.5*height*2.5];

double addX = 1.0 / 2.5;
double addY = 1.0 / 2.5;
double dx = 0.0; double dy = 0.0;
int x = 0; int y = 0;

while (dy < height)
{
  while (dx < width)
  {
    int gray = bic.getPixel(dx,dy);
    scaledImage[(y*width)+x] = (short)val;
  }
  dx = dx + addX;
  x++;
}
dy = dy + addY;
y++;
```

Zuletzt sei noch kurz die Klasse *ResultLog* erwähnt. Sie dient lediglich dazu, die während der Kepstrum-Filterung berechneten Laufzeit-Parameter sowie Verschiebungswerte in eine Datei zu schreiben. Der Name dieser Datei wird dem Konstruktor übergeben und ist als *Cepstrum.log* im Quelltext festgelegt. Das *ResultLog*-Objekt ist so konzipiert, daß das LogFile bei der Instanziierung geöffnet wird und nach Aufruf der Methode *append* zum Anhängen eines übergebenen Strings an das Dateiende wieder geschlossen wird. Da *ResultLog* sowohl von *CepstrumAnalysis* als auch von *Cepstrum* genutzt wird, kann so der exklusive Schreibzugriff auf einfache Weise hergestellt werden.

Abschließend ist wieder das Diagramm der in diesem Unterkapitel besprochenen Klassen zur Kepstrum-Filterung dargestellt, siehe Abbildung 9.5.

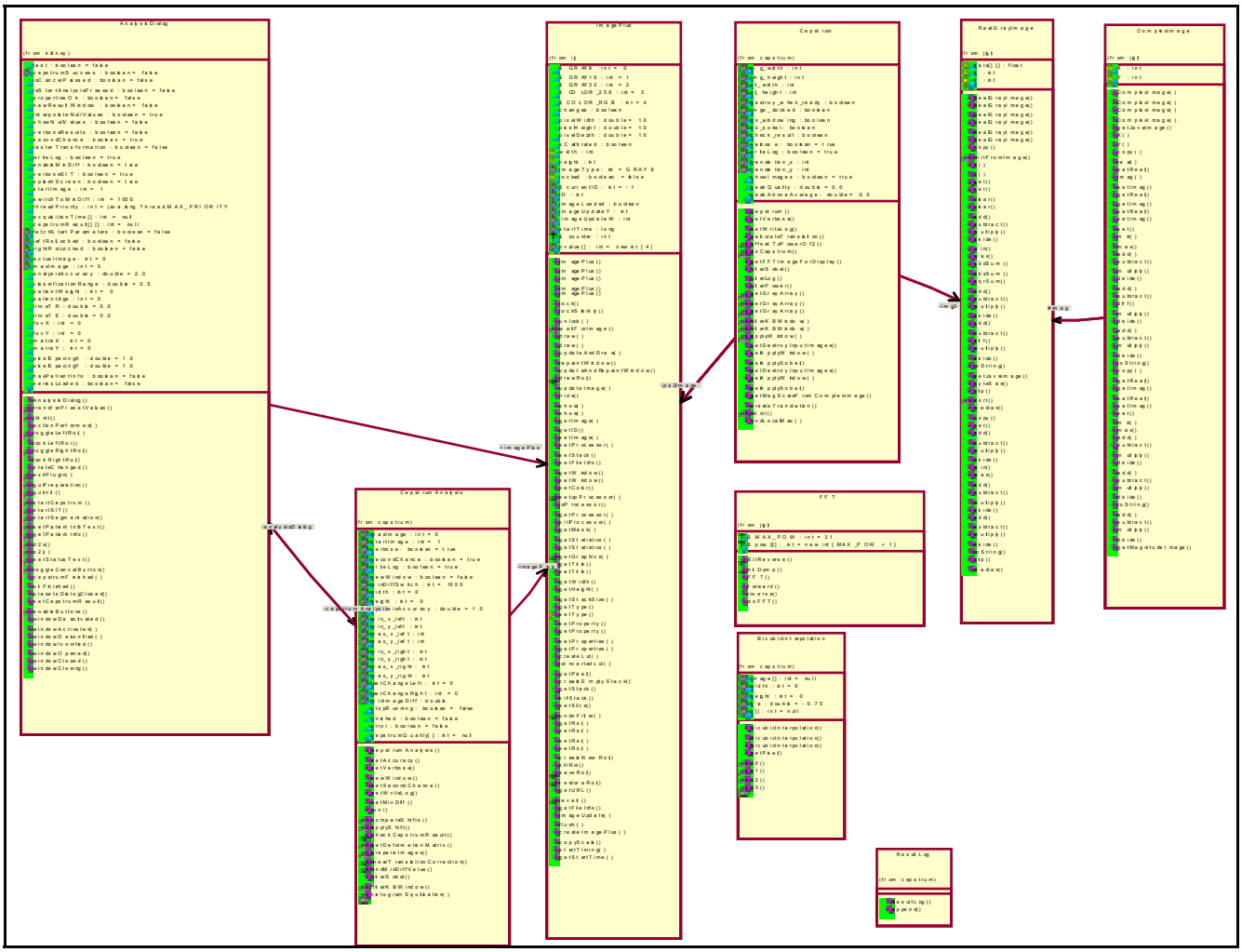


Abbildung 9.5: Die zur Kepstrum-Filterung benötigten Klassen.

9.7 Berechnung der SIT-Kurven

Die Berechnung der Signal-Intensitäts-Zeit-Kurven mit automatischer Klassifizierung der Nierengewebe setzt eine erfolgreiche Kepstrum-Filterung voraus. Die große Berechnungsklasse *SITAnalysis* ist für die Erstellung der Kurven sowie für die Klassifizierung verantwortlich. Die Klassen *SITResults*, *SITAdjuster*, *JLabelResults* und *ImageLabel* sind für die Darstellung und Justierung der Ergebnisse verantwortlich und werden in Kapitel 9.9 beschrieben. Eine manuelle Auswertung – ohne automatische Gewebeklassifizierung – wird ebenfalls in der Klasse *SITAnalysis* durchgeführt. Kapitel 9.8 geht auf die hierfür benötigten Methoden in *SITAnalysis* genauer ein.

9.7.1 Das Paket *sit*

Die Klasse *SITAnalysis* benötigt Zugriff auf die (korrigierte) Nierensequenz in *ImagePlus*, die qualitativen Bewertungen der Kepstrum-Filterung, die Aufnahmezeiten sowie linke und rechte ROI. Um nicht mit einer unübersichtlichen Anzahl *setSomething*-Methoden zu arbeiten, wird einfach eine Referenz auf das *AnalysisDialog*-Objekt (siehe Kapitel 9.5.1) im Konstruktor übergeben. Die benötigten Variablen und Objektreferenzen sind dort als *public* deklariert und bieten so einen einfachen Zugriff.

Schon im Konstruktor von *SITAnalysis* wird geprüft, ob im *ExImageCanvas* Haupt-ROIs für die rechte und/oder linke Gesamtniere eingezeichnet sind. Ist dies nicht der Fall, wird abgebrochen. Manchmal sind mehr als nur ein oder zwei Haupt-ROIs eingezeichnet, d.h. es wurden spezifische ROIs für

verschiedene Gewebearten in einem Bild oder mehreren Bildern der Sequenz markiert. Ist dies der Fall, hat der Benutzer – aus Sicht der Berechungsklasse – die Gewebeklassifizierung von Hand vorgenommen und die automatische Differenzierung wird für die entsprechende Bildhälfte nicht durchgeführt. Ebenso wird für die rechte und linke Bildhälfte geprüft, ob eine Kepstrum-Filterung überhaupt durchgeführt wurde und mit welchen Ergebnissen. Wurde die Filterung nicht durchgeführt, sind alle Bildhälften als ungültig für die Berechnung der SIT-Kurven markiert. Entsprechend sind alle Einträge im Array *manualSelector* der Klasse *ExImageCanvas* für jedes Halbbild der Sequenz auf **false** gesetzt. Eine Kurvenberechnung findet in diesem Fall nicht statt.

Wie schon die Klasse *CepstrumAnalysis*, so ist auch *SITAnalysis* von *java.lang.Thread* abgeleitet und implementiert die Methode *run*, die durch den Aufruf von *start* angestoßen wird. Alle Funktionalität zur Berechnung der SIT-Kurven ist hier untergebracht. Wie bereits mehrfach erwähnt, ist die Berechnung der Kurve für die Gesamtnieren recht einfach. Ausgehend von einer atmungskorrigierten Sequenz und einer entlang der Kontur der Niere eingezeichneten Haupt-ROI (vergleiche Abbildung 8.1) wird diese zunächst auf alle Bilder der Serie projiziert. Für alle innerhalb der ROI befindlichen Pixel wird der mittlere Grauwert und die Standardabweichung für jedes Bild der Sequenz berechnet und in einem *double[]*-Array abgelegt. Ist die ROI nicht rechteckig, sondern eine Freihand- oder Polygon-ROI, so lassen sich alle innerhalb der ROI befindlichen Pixel mit folgender Schleife feststellen:

```
Rectangle rect = leftROI.getBoundingBox();
int[] mask = roi.getMask();
if (mask != null)
{
for (x = 0; x < rect.width; x++)
{
  for (y = 0; y < rect.height; y++)
  {
    if (mask[(y*(rect.width))+x] != -1)
    {
```

Mit Hilfe der Methoden *derivation* und *getMaxArrayVal* wird nun das Maximum der ersten Ableitung bestimmt. Auch die weiteren Berechnungsschritte halten sich strikt an die in Kapitel 8 vorgestellte Theorie. Aufgrund des berechneten Wendepunktes können die vier Hilfsbilder berechnet werden, die intern als Arrays verwaltet werden, siehe auch Abbildung 8.2. Ebenso werden die in Kapitel 8.1.2 besprochenen (dynamischen) Parameter $d_{somewhat}$ und $g_{somewhat}$ anhand der aktuellen Sequenz bestimmt.

Nun sind alle zur automatischen Gewebeklassifizierung notwendigen Vorbereitungen getroffen. Die Methode *classifyPixels* führt letztendlich die Klassifizierung durch und schreibt ihre Ergebnisse im *int[]*-Array *pixClassification*. Zur Visualisierung werden die als Nierenmark berechneten Pixel rot und die Pixel der Nierenrinde gelb eingefärbt, siehe auch Kapitel 8.1.2.

Anhand der im Klassifizierungs-Array gespeicherten Werte können nun die Kurven für Mark und Rinde berechnet werden. Die Methode *calcPixelCurve* bekommt die gewünschte Bildhälfte und den Gewebetyp übergeben und führt die Berechnung durch.

Sollten einzelne Bilder der Sequenz bei der Kepstrum-Filterung nicht erfolgreich korrigiert worden sein, ist dies für die automatische Gewebeklassifizierung i.d.R. nicht hinderlich. Um Nullstellen in den berechneten SIT-Kurven zu vermeiden, werden die Werte für diese Bilder aus dem (zeitlichen) Vorgänger und Nachfolger durch Aufruf der Methode *interpolateNullValues* interpoliert.

Wie bereits erwähnt, startet die Kepstrum-Filterung nicht bei Bild eins, sondern erst bei Bild vier, siehe Kapitel 9.6. Um daher die ersten drei Positionen der berechneten Kurven mit plausiblen Ergebnissen zu belegen, werden durch die Methode *fillMissingValues* einfach die für Bild vier berechneten Werte auf die Positionen eins bis drei übertragen. Das gleiche Verfahren wird für die

letzten Werte der Kurve durchgeführt, wenn hier eine Folge von Bildern als ungültig für die SIT-Berechnung markiert sein sollte.

Alle weiteren Methoden dieser umfangreichen Klasse werden weitgehend bei der Ergebnisdarstellung in Kapitel 9.9 benötigt. In Abbildung 9.6 findet sich das Klassendiagramm zur Berechnung der SIT-Kurven.

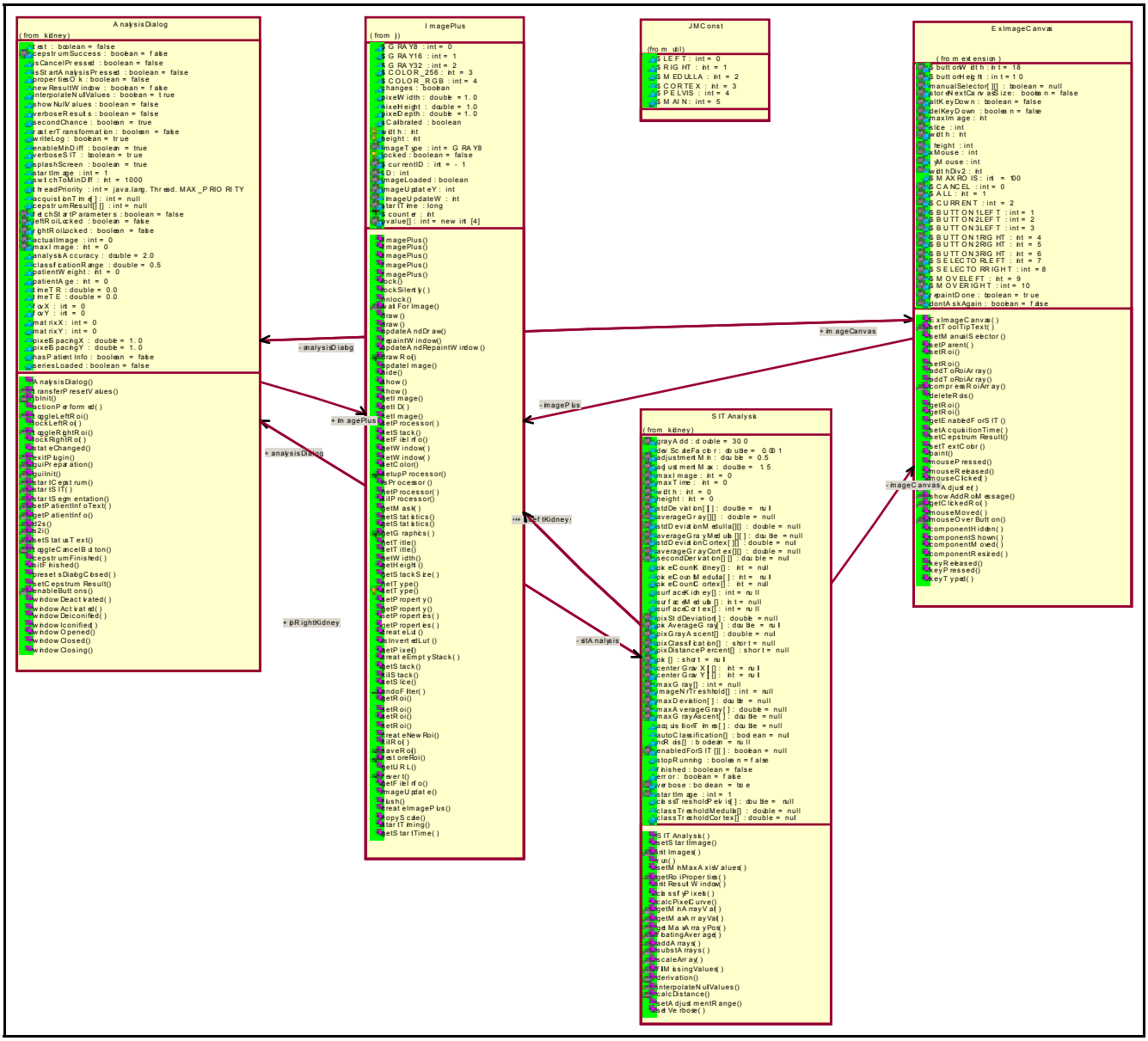


Abbildung 9.6: Klassendiagramm für die SIT-Kurven-Berechnung.

9.8 Manuelle Bearbeitung

Es hat sich bei der praktischen Anwendung gezeigt, daß Probleme bei der Kepstrum-Filterung zumeist aus unregelmäßigen Nierenanschnitten entstehen, vergleiche Kapitel 7.7.2. Aber auch eine überdurchschnittlich schnelle Kontrastmittelaufnahme der Niere kann, selbst unter Anwendung des in Kapitel 7.4 diskutierten Histogrammangleichs, Probleme bereiten. Aus dieser Erkenntnis ergab sich die Notwendigkeit, das System um die Möglichkeit einer manuellen Auswertung zu erweitern. Die funktionale Perfusionsuntersuchung aller Nierensequenzen ist so an einem Arbeitsplatz durchführbar.

Durch die Art der Bearbeitung einer Nierensequenz mittels Kepstrum-Filterung ergibt sich (subjektiv) ein fließender Übergang zwischen einem schlechten und einem guten Ergebnis. Werden nur einzelne Bilder nicht richtig korrigiert, hilft oftmals ein zweiter Durchlauf der Atmungskorrektur. Sollten trotzdem einzelne Bilder der Sequenz gegenüber den übrigen verschoben bleiben, obwohl das Ergebnis als insgesamt gut bezeichnet werden kann, so können diese „Aussetzer" manuell justiert werden. Dabei bietet das System die Möglichkeit, einzelne Bildhälften in 0,1- bis 2,0-Pixelschritten zu verschieben. Die Klasse *ManualMover* implementiert die dazu notwendige Funktionalität. Wird durch diese Art der Nachjustierung die gleiche Kongruenz der Nieren wie bei einer optimalen Durchführung der Kepstrum-Filterung erzielt, so steht einer automatischen Gewebeklassifizierung nichts im Wege.

Aus verschiedenen Gründen, z.B. bei sehr schlechten Kepstrum-Ergebnissen, kann aber auch eine herkömmliche, komplett manuelle Auswertung gefragt sein. Hierzu ist dann das Einzeichnen mehrerer ROIs für die verschiedenen Gewebearten sowie die Gesamtniere in jedem Bild der Sequenz notwendig. Eine automatische Gewebeklassifizierung findet dann nicht mehr statt. Die Klassen *ExImageCanvas* und *SITAnalysis* wurden bereits in den vorigen Kapiteln 9.5.2 und 9.7.1 vorgestellt, jedoch wird ihre Funktionalität hinsichtlich der manuellen Auswertung hier gesondert erklärt.

9.8.1 Das Paket *kidney*

Die letzte noch nicht diskutierte Klasse dieses Paketes ist *ManualMover*. Wie einleitend bereits erwähnt, dient sie dem Verschieben von einzelnen Halbbildern in vom Benutzer einstellbaren (nicht ganzzahligen) Pixelschritten. Ein *ManualMover*-Objekt implementiert einen kleinen Dialog, auf dem der Benutzer die gewünschten Pixelschritte mit einem Schieberegler einstellen kann, sowie vier Buttons für die Verschiebungsrichtungen. Zur Berechnung des verschobenen Halbbildes wird die schon vorgestellte Klasse *BicubicInterpolation* benutzt, siehe Kapitel 9.6.2. Bei jedem Klick auf eine der Richtungsschaltflächen wird *applyShift* mit den *x*- und *y*-Verschiebungswerten als Parameter aufgerufen. Die Berechnung der Verschiebung für ein Halbbild (128×256 Pixel) dauert weniger als 0,5 Sekunden und wird sofort im *Canvas* neu eingezeichnet. Die Klasse selbst wird durch einen Klick auf die Verschiebungs-Schaltfäche im *ExImageCanvas* aufgerufen, vergleiche Kapitel 10.2.5.

9.8.2 Das Paket *extension*

Hier soll noch einmal auf die Klasse *ExImageCanvas* eingegangen werden. Sie bietet die Möglichkeit, für jedes Halbbild der Sequenz eine Haupt-ROI sowie für die beiden Gewebetypen Medulla und Cortex jeweils bis zu 100 Unter-ROIs einzuzeichnen. Der Benutzer muß dabei allerdings selber festlegen, welches Gewebe sich innerhalb der gerade eingezeichneten ROI befindet. Zu diesem Zweck befinden sich am oberen Rand des *Canvas* in jedem Halbbild drei farbige Schaltfächen:

- Grün: setzt die Haupt-ROI für das aktuelle Halbbild fest
- Gelb: setzt eine Cortex-ROI fest
- Rot: setzt eine Medulla-ROI fest

ExImageCanvas implementiert einen *MouseListner*, der bei jedem Klick die Position des Mauszeigers abfragt und die gedrückte Schaltfäche berechnet. Da eine ROI entweder nur für das aktuelle Halbbild oder aber für alle Halbbilder der Sequenz festgesetzt werden kann (also auf alle anderen Halbbilder projiziert wird), öffnet sich bei dem Klick auf einen der Buttons ein kleiner, abschaltbarer Dialog, der genau diese Information abfragt, siehe Abbildung 10.21. Ist der Dialog einmal abgeschaltet, kann der Benutzer durch ein Drücken der ALT-Taste während der Klicks auf eine der Schaltflächen bewirken, daß er die gerade festgesetzte ROI auf alle Bilder der Sequenz projiziert wird. Daher implementiert die Klasse auch einen *KeyListener*, der die gedrückte Taste abfragt. Die überladene Methode *setRoi* setzt die Haupt-ROI, entweder nur für die übergebene Bildnummer oder für die gesamte Sequenz. Gleiches bewirkt die Methode *addRoiToArray* für die Unter-ROIs für Cortex und Medulla, jedoch wird hier zusätzlich das gewünschte Gewebe als Parameter mit übergeben.

9.8.3 Das Paket *sit*

Auch hier soll nochmals auf eine schon beschriebene Klasse eingegangen werden. Die Berechnungsklasse *SITAnalysis* stellt bereits im Konstruktor fest, ob für jedes Halbbild eine Haupt-ROI eingezeichnet ist und ob eventuell Unter-ROIs markiert sind. Sollte die Haupt-ROI sich nicht in jedem Bild an identischer Position befinden oder sollten Unter-ROIs eingezeichnet sein, so findet keine automatische Gewebeklassifizierung statt.

Die Auswertung der Gesamt-ROI zur Erstellung der Kurve für die Gesamt-Niere kann wie in Kapitel 8.1.1 besprochen durchgeführt werden. Die Unter-ROIs werden hinsichtlich ihres Gewebetyps zusammengefaßt und berechnet. Sind z.B. fünf ROIs für die Nierenrinde markiert, werden alle innerhalb der ROIs befindlichen Pixel zu einer Pixelmenge zusammengefaßt, auf der dann die Berechnung des mittleren Grauwertes und der Standardabweichung durchgeführt wird.

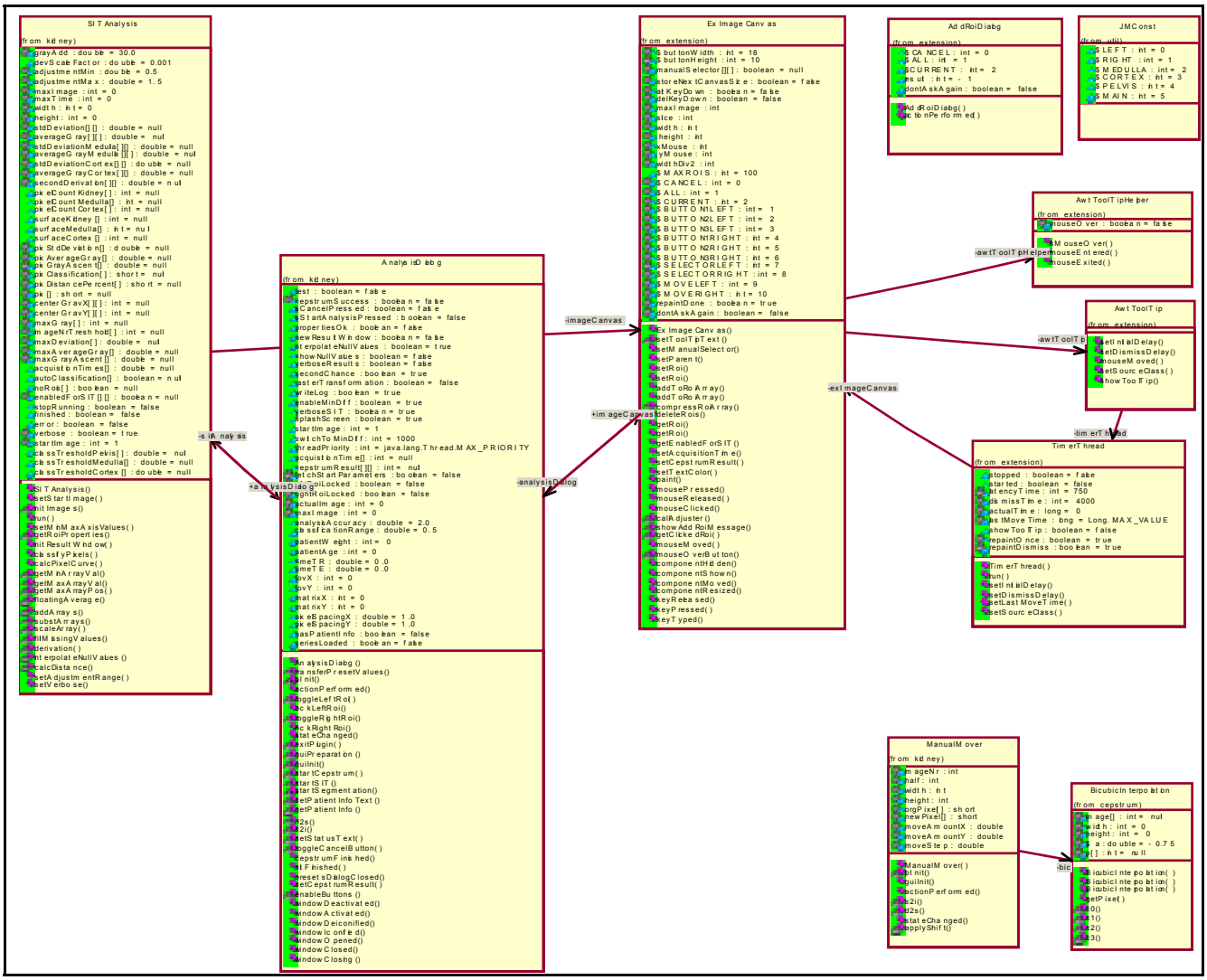


Abbildung 9.7: Darstellung der zur manuellen (Nach-)Bearbeitung benötigten Klassen.

9.9 Ergebnisdarstellung, Justierung und Export

Die Klasse *SITAnalysis* speichert die 12 zu berechnenden SIT-Kurven in Arrays vom Datentyp *double* (rechts / links - Rinde / Mark / Gesamtniere - mittlerer Grauwert / Standardabweichung). Die Länge der Arrays wird durch die Anzahl der beim DICOM-Import ausgewählten Bilder bestimmt, in der Praxis

ein Wert zwischen 30 und 60. Alle für die Ergebnisdarstellung und –justierung benötigten Klassen sind ebenfalls im Paket *sit* enthalten.

9.9.1 Das Paket *sit*

Die Klasse *SITResults* implementiert das Hauptfenster zur Darstellung der Ergebnisse. Sie bekommt im Konstruktor eine Referenz auf die Berechnungsklasse *SITAnalysis* übergeben und erhält so Zugriff auf die dort berechneten Kurvendaten. Außerdem werden von der Methode *classifyPixels* in *SITAnalysis* zwei bildliche Darstellungen der Ergebnisse berechnet. Dies sind zwei *ImagePlus*-Objekte mit je einem Bild, welches den gewählten (rechteckigen) Ausschnitt um die Niere enthält sowie die eingefärbten Pixelklassifizierungen, siehe Kapitel 8.1.2. Diese werden ebenfalls dem Konstruktor übergeben und durch die Hilfsklasse *ImageLabel* in die GUI eingefügt. Die Darstellung der Kurven wird ebenfalls von einer Hilfsklasse übernommen, der Klasse *JLabelResults*. Ein *JLabelResult*-Objekt übernimmt dabei die Darstellung aller Kurven eines Gewebetyps (ganze Niere, Mark, Rinde) und stellt die Kurve für linke und rechte Niere in einem gemeinsamen Koordinatensystem dar. Insgesamt werden also drei *JLabelResult*-Objekte benötigt. *SITResults* steuert über Kontrollkästchen die Darstellungsform der Kurven auf den *JLabelResults*. Der Benutzer kann folgende Optionen wählen:

- Darstellung der Standardabweichung an- / ausschalten
- Kurven glätten
- Kontrollpunkte an- / ausschalten

Das Glätten der Kurven geschieht in der Methode *smoothCurves* bzw. *floatingAverage*. Die gewünschte Darstellungsform wird durch die Hilfsmethoden *showMarks* und *showStandardDeviation* in *JLabelResults* an- oder ausgeschaltet.

Die Methode *Export* in *SITResults* wird bei Klick auf den entsprechenden Button aufgerufen. Sie fordert den Benutzer auf, Zielverzeichnis und Dateiname anzugeben und exportiert die berechneten SIT-Werte dann als Textdatei. Die Daten werden dabei mit einem Tabulator-Zeichen als Delimiter versehen, um den einfachen Import in Tabellenkalkulations- oder Datenbankprogramme zu gewährleisten. Die Textdatei enthält einen Kopf mit Patienteninformationen, darunter tabellarisch die Berechnungswerte in 12 Spalten sowie eine zusätzliche Spalte für den Aufnahmezeitpunkt. Das Ergebnis eines Imports mit MICROSOFT EXCEL sieht wie folgt aus:

Renographie												
Patient:			PatientName			FOV (mm):			300	300		
ID:			999999			Matrix:			256	256		
Untersuchungsdatum:			31.04.1999			Pixelfläche (mm²):			1.373291			
Geburtsdatum:			31.09.1999									
NAST (%, Seite):			*									
Geschlecht:			F									
Gewicht (kg):			50									
NaCl-Infusion (ml):			*									
Magnevist (ml):			*									
Originaldaten												
	Gesamtniere rechts		Gesamtniere links		Nierenrinde rechts		Nierenrinde links		Nierenmark rechts		Nierenmark links	
	ROI	Pixel	ROI	Pixel	ROI	Pixel	ROI	Pixel	ROI	Pixel	ROI	Pixel
	2319	1689	2539	1849	494	360	427	311	1134	826	1137	828
Zeit	Signalintensität		Signalintensität		Signalintensität		Signalintensität		Signalintensität		Signalintensität	
(s)	Mittel	Stabw	Mittel	Stabw	Mittel	Stabw	Mittel	Stabw	Mittel	Stabw	Mittel	Stabw
0.0	24.6	18.9	32.9	21.5	14.1	10.8	23.7	18.1	18.5	16.0	25.4	20.8
2.4	24.6	18.9	32.9	21.5	14.1	10.8	23.7	18.1	18.5	16.0	25.4	20.8
4.5	24.6	18.9	32.9	21.5	14.1	10.8	23.7	18.1	18.5	16.0	25.4	20.8
6.6	24.6	18.9	32.9	21.5	14.1	10.8	23.7	18.1	18.5	16.0	25.4	20.8
8.7	25.1	18.6	33.2	21.1	16.6	13.5	28.0	20.3	18.3	15.6	24.9	19.9
10.8	24.4	18.0	33.4	21.1	18.5	15.3	34.7	25.3	17.1	14.0	24.9	19.5
12.9	24.6	18.3	33.3	21.1	18.7	15.0	36.1	26.7	17.1	14.1	24.5	19.5
14.9	24.0	18.3	33.3	21.0	17.5	15.3	34.3	24.7	17.0	14.3	24.8	19.8

Tabelle 9.1: Import der vom System exportierten Berechnungsdaten mit einem Tabellenkalkulationsprogramm.

Durch Klick auf die *Adjust Results*-Schaltfläche wird ein Objekt der Klasse *SITAdjuster* erzeugt, mit dessen Hilfe die Klassifizierungs-Ergebnisse visuell angepaßt werden können. Zunächst soll aber kurz auf die von *SITResults* verwendeten Hilfsklassen *ImageLabel* und *JLabelResults* eingegangen werden.

Einem *ImageLabel*-Objekt kann im Konstruktor direkt ein *java.awt.Image*-Objekt übergeben werden. Einmal instanziiert und innerhalb eines Fensters als GUI-Objekt angezeigt, kann das dargestellte Bild jederzeit mit *setImage* neu gesetzt werden und wird sofort (automatisch) aktualisiert. Allerdings ist der rechteckige Ausschnitt um die Niere zur Ergebnisdarstellung etwas zu klein, deswegen wird er zunächst vergrößert:

```
java.awt.Image scaledImage = null;
java.awt.Image originalImage = imp.getImage();
  // Originalbild liegt als ImagePlus-Objekt vor
int x = (int)((double)imp.getWidth() * scale);
int y = (int)((double)imp.getHeight() * scale);
scaledImage =
  originalImage.getScaledInstance
    (x,y,originalImage.SCALE_SMOOTH);
mediaTracker.addImage(scaledImage, 0);
mediaTracker.waitForAll();
```

Danach wird *scaledImage* dann an das *ImageLabel*-Objekt übergeben.

Die Klasse *JLabelResult* ist von der SWING-Klasse *JLabel* abgeleitet und stellt die Ergebniskurven in einem Koordinatensystem dar. Durch die Methoden *setStdDeviation* und *setAverageGray* werden die Arrays mit den berechneten Werten übergeben. Mit *setXScale* und *setYScale* werden die Wertebereiche für die Achsen des dargestellten Koordinatensystems festgelegt. Über die Methoden *showMarks* und *showStandardDeviation* können die schon besprochenen Formatierungen gesetzt werden. Das Glätten der Kurven wird in *SITResults* durchgeführt und die dargestellten Kurvenwerte dann überschrieben. Die Anzeige aktualisiert sich automatisch. Alle weiteren Funktionen sind zum Aufbau der Grafik benötigte Hilfsfunktionen. Zusätzlich wird noch ein *MouseListener* implementiert, der dem Benutzer die Möglichkeit bietet, mit der Maus auf eine bestimmte Stelle der Kurve zu zeigen, um die Meßwerte für diesen Zeitpunkt zu erfahren. Abbildung 9.8 zeigt ein Beispiel eines *JLabelResults*.

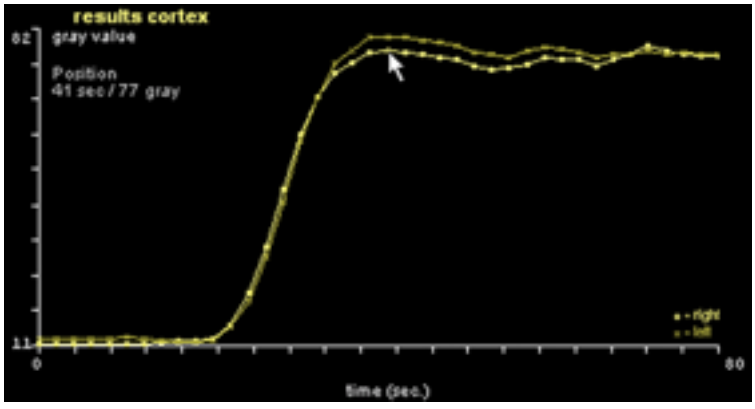


Abbildung 9.8: Ein *JLabelResult*. Links oben werden die Werte für die Kurve an der Mausposition angezeigt.

Die Klasse *SITAdjuster* wird aus der Klasse *SITResults* bei Klick auf die entsprechende Schaltfläche instanziiert und dargestellt. Sie bietet die Möglichkeit, die Ergebnisse der automatischen Gewebeklassifizierung nachträglich zu justieren. Die Abbildung 10.20 im folgenden Kapitel zeigt den von *SITAdjuster* dargestellten Dialog. Für linke und rechte Niere wird jeweils das Bild von der Position der Sequenz dargestellt, das dem berechneten Wendepunkte der Kurve für das Gesamtorgan entspricht,

vergleiche Kapitel 8.1.1. Direkt daneben wird das eingefärbte Klassifi-zierungsergebnis abgebildet. In beiden Fällen wird ein *ImageLabel* mit 2-facher Vergrößerung zur Darstellung benutzt. Dem Bearbeiter werden für rechte und linke Niere jeweils drei justierbare Schwellenwerte angeboten, die mittels Schiebereglern verändert werden können. Die Werte werden auf einen vorgegebenen Zahlenbereich [0,5...1,5] normiert. Die Intervallgrenzen sind im *PresetsDialog* bis auf [0,2...1,8] erweiterbar, siehe auch Kapitel 10.2.8. Indem *SITAdjuster* im Konstruktor eine Referenz auf die Berechnungsklasse *SITAnalysis* übergeben wird, ist ein direkter Zugriff auf die dort gespeicherten Schwellenwerte möglich. Bei jeder Veränderung eines der Schieberegler werden diese nun neu gesetzt und die Methode *classifyPixels* in *SITAnalysis* aufgerufen. Die neu berechnete Klassifizierung wird während der Berechnung auf die eingefärbten Klassifizierungsbilder übertragen und direkt dargestellt. Einen schnellen Prozessor vorausgesetzt[19], ergibt sich somit ein Echtzeit-Klassifizierungs-Algorithmus, mit dem der Benutzer die veränderten Schwellenwerte direkt visuell kontrollieren kann.

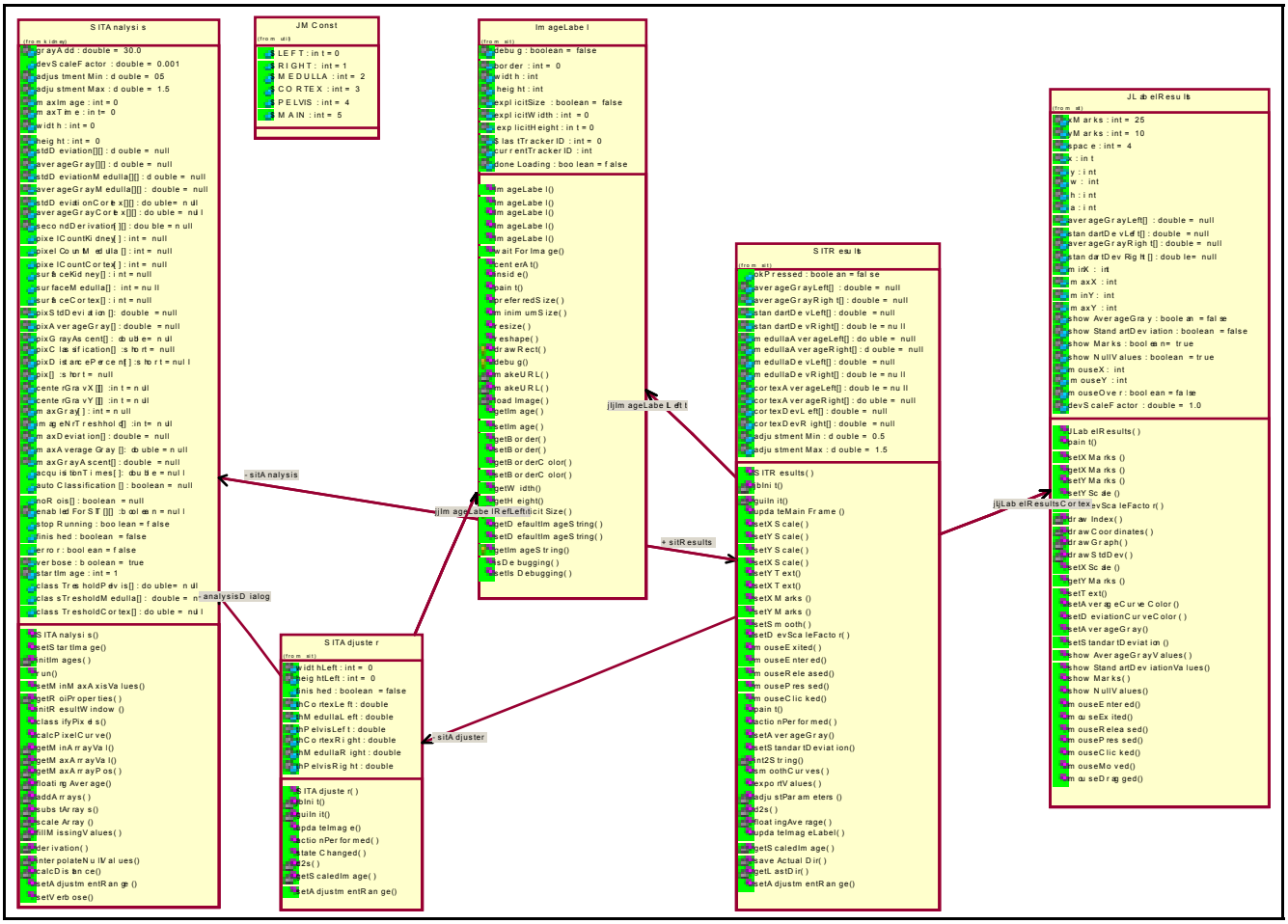


Abbildung 9.9: Die zur Ergebnisdarstellung benötigten Klassen.

9.10 Weitere Pakete

Die beiden im folgenden beschriebenen Pakete *segmentation* und *scaler* sind eher als „Nebenprodukte“ der Programmierung zu sehen. Wie in Kapitel 7.8 beschrieben, wurde anfangs versucht, die Kontur der Niere mit Hilfe eines Segmentierungsverfahrens zu erkennen, um so dem Benutzer das Einzeichnen einer ROI zu ersparen. Die Funktionalität des dabei entwickelten Segmentierungsalgorithmus läßt sich je nach Bildmaterial von gut bis ungenügend bewerten. Jedoch wurde die Klasse zur Segmentierung nicht verworfen, sondern mit in das Programm integriert. Der

[19] Ein PENTIUM II mit 300 MHz oder besser wird empfohlen.

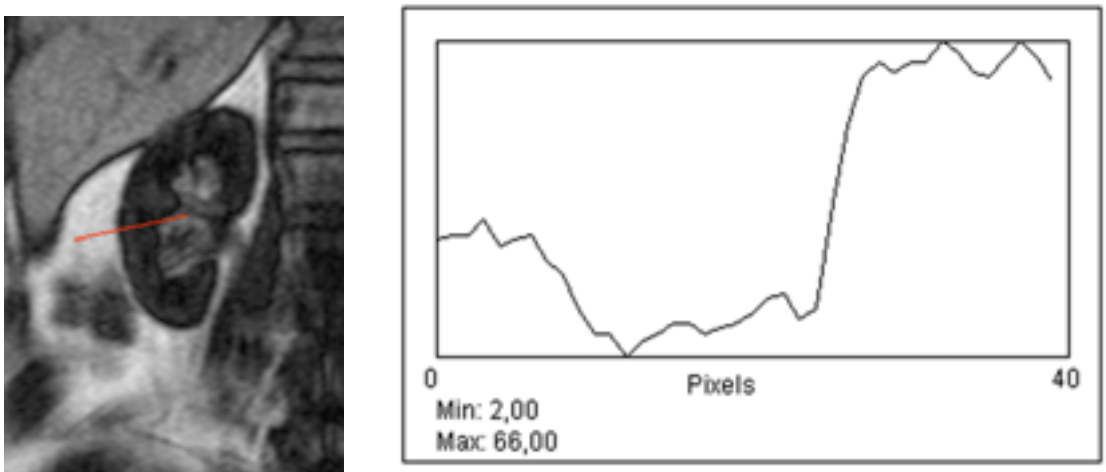
Aufruf geschieht vom Benutzerdialog der Klasse *AnalysisDialog* aus. Die entsprechende Schaltfäche für den Aufruf wurde vorsichtig mit *Try Segmentation* betitelt, um dem Benutzer angesichts der sehr unterschiedlichen Resultate nicht allzu viel zu versprechen.

Weiterhin wurde die Klasse *BicubicInterpolation* benutzt, um ein zusätzliches Skalierungs-*Plugin* für IMAGEJ zu implementieren. Dafür mußten lediglich die eine Aufrufklasse und ein Dialog für das *Plugin* geschrieben werden, die sich im Paket *scaler* befinden.

9.10.1 Das Paket *segmentation*

Die einzige Klasse dieses Paketes heißt *KidneyExtraction*. Sie bekommt im Konstruktor eine Referenz auf das *ImagePlus*-Objekt übergeben; die Segmentierung findet dabei immer in der aktuell angezeigten Schicht (= Bildnummer der Sequenz) statt. Einzige Voraussetzung ist, daß der Benutzer eine rechteckige ROI um eine der Nieren markiert hat, was aber schon in der Methode *startSegmentation* in *AnalysisDialog* abgefragt wird. Durch Aufruf der Methode *doSegmentation* wird die Detektion der Nierenkontur dann gestartet.

Zunächst wird nun das Eingabebild mit einem Median-Filter geglättet, vergleiche Kapitel 5.2.3. Die Segmentierung findet nun mittels eines „Scanline“-Verfahrens statt. Ausgehend vom Mittelpunkt der eingezeichneten (rechteckigen) ROI werden im Uhrzeigersinn in 3°-Schritten Scanlines zum Rand der ROI erzeugt. Entlang der Scanline wird das Grauwert-Intensitätsprofil durch die bereits in der *ImageProcessor*-Klasse von IMAGEJ implementiert Methode *getLine* berechnet. Abbildung 9.10 zeigt eine eingezeichnete Scanline mit dem dazugehörigen, von IMAGEJ berechneten Intensitätsprofil.



(a) **(b)**

Abbildung 9.10: Markierte Scanline, rot (a), Intensitätsprofil der Scanline (b), ausgehend vom Mittelpunkt der Niere.

Wie bereits in Kapitel 3.2.1 und 7.8 beschrieben, erzeugt die Turbo-Flash-Aufnahmetechnik eine dunkle Kontur rund um die Niere, so daß nach dem Punkt mit der niedrigsten Intensität im Profil gesucht werden kann.

Um zusätzliche Sicherheit zu gewinnen, wird ein Kantenbild berechnet und auch auf diesem eine Scanline erzeugt. Da der in Kapitel 5.2.4 beschriebene Sobel-Filter Schwächen bei diagonalen Kantenverläufen hat, wird hier eine Implementierung des Kompaß-Filters angewendet. Die Methode *getRotatedSobel* liefert ein entsprechend der als Parameter übergebenen Grad-Zahl optimiertes Kantenbild. Dabei arbeitet die Methode mit 8 verschiedenen Kernel, einem für jede Himmelsrichtung. Im sobel-gefilterten Bild ergeben sich bei Kanten sehr helle Intensitäten, siehe auch Abbildung 5.8b. In der Scanline wird also nach dem Maximum gesucht. Stimmen die berechneten Punkte in beiden Scanlines überein, so wird ein Konturpunkt erzeugt. Die berechnete Kontur wird dabei als

java.awt.Polygon-Datentyp abgespeichert und zum Ende der Berechnung in eine IMAGEJ-ROI umgewandelt:

```
Roi roi = new PolygonRoi
  (poly.xpoints, poly.ypoints, poly.npoints, imagePlus,false);
imagePlus.setRoi( roi);
```

Ist der erste Umlauf der beiden Scanlines fertig (120 mal 2 Linien in 3°-Abständen), wird in einem zweiten Umlauf versucht, die Kontur zu glätten. Dabei können als Kriterien ausgenutzt werden, daß ein durch drei Konturpunkte festgelegter Winkel nicht zu groß werden darf (im Mittel < 35°) sowie der Abstand zweier benachbarter Konturpunkte nicht zu groß sein darf (im Mittel < 4 Pixel). Die Hilfsmethoden *calcAngle* und *calcDistance* werden für die Berechnung der Kriterien benutzt. Konturpunkte aus dem ersten Durchlauf, die den Test nicht bestehen, werden entweder korrigiert oder ganz verworfen, falls eine Korrektur kein sinnvolles Ergebnis bringt.

Bevor das Kontur-Polygon wie oben beschrieben in eine ROI umgewandelt wird, kann es mit der Methode *smoothPolygon* nochmals geglättet werden.

9.10.2 Das Paket *scaler*

Die in diesem Paket enthaltenen Klassen *Bicubic_Scaler* und *ScaleDialog* ermöglichen den Einsatz der in der Klasse *BicubicInterpolation* implementierten bikubischen Interpolation direkt als IMAGEJ-*Plugin*. Die Klasse *Bicubic_Scaler* implementiert die *Plugin*-Schnittstelle von IMAGEJ und prüft zunächst den Typ des zu skalierenden Bildes. Wie bereits erwähnt, können mit *BicubicInterpolation* 8-, 16-, und 32-Bit-Grauwertbilder skaliert werden. Liegt ein anderer Bildtyp vor oder ist kein Bild geöffnet, wird abgebrochen. Ansonsten wird eine Instanz der Klasse *ScaleDialog* erzeugt, die den Benutzerdialog zur Einstellung der Skalierungswerte bereitstellt. Dabei können die neuen Bilddimensionen entweder per Tastatureingabe in Textfelder oder durch die Benutzung von Größer/Kleiner-Schaltflächen eingestellt werden. Außerdem kann der Benutzer wählen, ob er die aktuellen Bildproportionen (Höhe/Breite) beibehalten will oder die neue Höhe und Breite unabhängig voneinander einstellen will. Die Berechnung des skalierten Bildes wird von der im Paket *cepstrum* enthaltenen Klasse *BicubicInterpolation* durchgeführt. Deren Funktionsweise wurde bereits in Kapitel 9.6.2 erläutert.

10 Installation und Bedienung

In diesem Kapitel werden alle notwendigen Schritte für die Inbetriebnahme und Bedienung des Systems besprochen. Die beiliegende CD enthält alle hier besprochenen Programme und Beispielsequenzen. Dieses Kapitel findet sich auch gesondert in einer Datei *Anleitung.pdf* im Verzeichnis *Manual* auf der CD. Damit kann der Benutzer nur das Handbuch ausdrucken und muß nicht immer auf den gebundenen Druck zurückgreifen, um Fragen zur Bedienung zu klären.

Die Bedienungsanleitung wird wieder entsprechend des Bearbeitungsablaufs einer Nierensequenz dargestellt, wie bereits in den Kapiteln 4 und 9 geschehen. Anhand der recht ausführlich erklärten Schritte sollten auch weniger computer-versierte Benutzer mit dem Programm zurechtkommen.

Die Bestandteile der Benutzeroberfläche (Fenster, Schaltflächen, etc.) sind in Englisch beschriftet.

Abbildung 10.1: Titelbild des IMAGEJ-*Plugins* zur renalen Funktionsanalyse.

10.1 Installation

10.1.1 Hard- und Softwarevoraussetzungen

Wie allgemein bei Bildverarbeitungssystemen, so sind auch hier die Ansprüche an die Hardware recht hoch. IMAGEJ und insbesondere das *Plugin* zur renalen Funktionsanalyse beanspruchen viel Rechenleistung und setzen ein schnelles System voraus, um in angemessener Zeit die Analysen durchführen zu können. Die Kepstrum-Filterung zur Korrektur der Atmungsbewegungen und die dabei durchzuführenden Fourier-Transformationen benötigen dabei die meiste Rechenzeit. Zwar ist die Analyse auf jedem Rechner mit einem PENTIUM Prozessor lauffähig, es wird jedoch dringend ein PENTIUM II mit mindestens 300 MHz und 64 MB Hauptspeicher empfohlen. Erst unter diesen Voraussetzungen läuft der zur manuellen Ergebnis-Nachjustierung eingesetzte Echtzeit-Algorithmus gut. Auf einem PENTIUM III mit 550 MHz und 256 MB Hauptspeicher benötigt die Kepstrum-Filterung für 40 Bilder mit doppelter Genauigkeit (Berechnung der Translation auf 0,5 Pixel) ca. 10 Minuten. Der gleiche Testlauf auf einem PENTIUM III mit 800 MHz und 384 MB Hauptspeicher[20] benötigte nur noch ca. 6 Minuten.

Soll das System zur medizinischen Diagnoseunterstützung eingesetzt werden, so sollte man die dadurch erzielten Zeiteinsparungen für den Diagnostiker berücksichtigen, siehe auch Kapitel 4. Eine Mehrausgabe für die Anschaffung eines sehr schnellen Rechners kann sich so, je nach Häufigkeit der Systemanwendung, schon nach wenigen Wochen rentieren.

Um die beiliegende CD verwenden zu können, muß am Rechner ein CD-ROM-Laufwerk angeschlossen sein. Sollen – zu Demonstrationszwecken – die Nierenserien auch von der CD geladen werden, so wird ein Laufwerk mit mindestens 20-facher Geschwindigkeit empfohlen. In der Regel werden die Daten aber lokal am Arbeitsplatz oder über ein Netzwerk verfügbar sein.

Das notwendige Betriebsystem ist nur insofern einzuschränken, als daß eine JAVA-VM[21] dafür vorhanden sein muß. Das System wurde unter WINDOWS95, WINDOWS98, WINDOWSNT 4.0, LINUX[22] und MACOS 8.5 erfolgreich getestet.

10.1.2 Installation und Programmstart

Für einen ersten Start des Programms zur renalen Funktionsanalyse unter einem MICROSOFT-WINDOWS Betriebssystem ist es zunächst erforderlich, einen kleinen Installationsvorgang durchzuführen. Dabei muß lediglich das Programmverzeichnis *RenalFunctionAnalysis* (mit allen Unterverzeichnissen) an beliebiger Stelle auf die Festplatte kopiert werden. Das Programmverzeichnis enthält bereits die zum Ausführen notwendige *Virtual Machine* für WINDOWS-Betriebssysteme. Lediglich das Attribut „s*chreibgeschützt*“ der kopierten Dateien muß vor dem ersten Start noch entfernt werden. Da sich die Dateien vorher auf einer CD befanden, ist es standardmäßig immer gesetzt. Dazu wechselt man in das Verzeichnis auf der Festplatte, welches das Ziel des vorherigen Kopiervorgangs war. Ein Doppelklick auf das kleine Hilfsprogramm „*ChangeFileAttributes*“ erledigt das Setzen der Dateiattribute automatisch. Nun sind die Vorbereitungen abgeschlossen, und das System kann durch einen Doppelklick auf das Symbol *RenalFunctionAnalysis* gestartet werden. Wie bereits erwähnt, werden mit dessen Hilfe die Nierensequenzen importiert und die renale Funktionsanalyse durchgeführt. Eine detaillierte Bedienungsanleitung für IMAGEJ und die *Plugins DICOM_import* und *Renal Function Analysis* finden Sie in Kapitel 10.2.

Auf der CD befinden sich im Verzeichnis *Examples* 12 anonymisierte Nierenserien. Zur einfacheren Einarbeitung empfiehlt es sich, zunächst diese Sequenzen als Test zu nutzen. Weiterhin befinden sich noch fünf korrigierte Sequenzen im Verzeichnis *Examples* / *CepstrumFiltered*. Sie können dazu

[20] Danke an LS-Bit (Bochum) für die Bereitstellung des Rechners zu Testzwecken.
[21] VM = Virtual Machine
[22] Kernel Version 2.2.13 (Distribution S.u.S.E. Version 6.3)

herangezogen werden, die Berechnung der SIT-Kurven auszutesten, ohne vorher die langwierige Korrektur der Atmungsbewegung durchführen zu müssen.

Auf einem anderen System als MICROSOFT-WINDOWS System läßt sich die renale Funktionsanalyse nicht auf die beschriebene Art und Weise starten. Identisch sind der beschriebene Kopiervorgang sowie das Löschen der Dateiattribute „Schreibgeschützt“. Falls noch keine JAVA-Laufzeitumgebung auf dem System installiert ist, so muß dies als nächster Schritt getan werden. Auf der beiliegenden CD befinden sich die JAVA 1.1.8-Laufzeitumgebungen im Verzeichnis *JavaVM*. Für die Betriebssysteme WINDOWS, LINUX, MACOS und SOLARIS existieren jeweils gleichnamige Unterverzeichnisse, in denen sich die Installation der Laufzeitumgebung befindet. Für den Fall eines MACOS-Betriebssystems wechselt man z.B. in das Verzeichnis *JavaVM / MacOS* und startet die dortige Installations-Routine *MRJ_2.2_Installer.sit*.

Das genaue Vorgehen des Programmstarts ist nun vom Betriebssystem und der gewählten VM abhängig und kann daher hier nicht explizit für jedes System beschrieben werden. Zur Erleichterung steht im Verzeichnis *RenalFunctionAnalysis* die *jar*-Datei mit den Namen *ij.jar* zur Verfügung. In diesem Archiv sind alle notwendigen Dateien für das Programm IMAGEJ sowie alle *Plugins* enthalten und können von jeder VM gestartet werden.

10.2 Bedienungsanleitung

10.2.1 IMAGEJ

Nach dem Start findet der Benutzer zunächst das Fenster des Bildverarbeitungsprogramms IMAGEJ, siehe Abbildung 10.2. Wie bereits im Umsetzungskapitel beschrieben, ist die renale Funktionsanalyse als sogenanntes *Plugin* für dieses Programm implementiert, ebenso wie das Modul zum Import der Nierensequenzen. Im Rahmen dieser Arbeit ist deshalb nur der IMAGEJ-Menüpunkt *Plugins* von Interesse. Die weiteren Menüs und deren Einträge sind natürlich nicht deaktiviert. Es steht dem Benutzer frei, sämtliche Bildverarbeitungsoperatoren von IMAGEJ auch an den Nierensequenzen auszuprobieren. Die sehr umfangreiche Funktionalität des Programms soll hier aber nicht weiter beschrieben werden. Auf der CD ist im Verzeichnis *JavaDOC / ImageJ* die Hilfe (und API) zu IMAGEJ in HTML-Form enthalten. Für weitere Informationen zu IMAGEJ siehe bitte [RASB98].

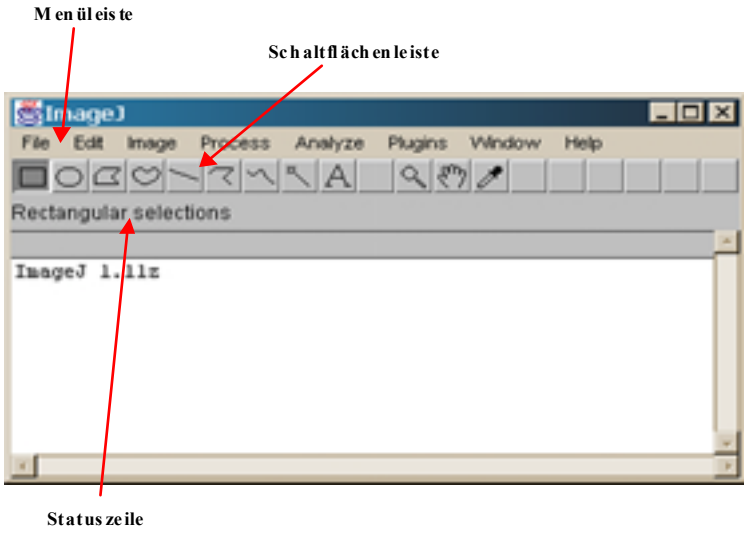


Abbildung 10.2: IMAGEJ-Hauptfenster.

10.2.2 Import der Nierensequenzen

Um eine Nierensequenz zu importieren, muß aus dem Menü *Plugins* der Menüpunkt *DICOM Import* aufgerufen werden. Es erscheint der Dialog zur Steuerung des Datenimports, wie in Abbildung 10.3 dargestellt. Zum Einlesen der DICOM-Dateien müssen jetzt die Auswahlkästchen *Include DICOM header* und *Load multiple images* aktiviert sein; diese sollten beim Aufruf des *Plugins* standardmäßig selektiert sein.

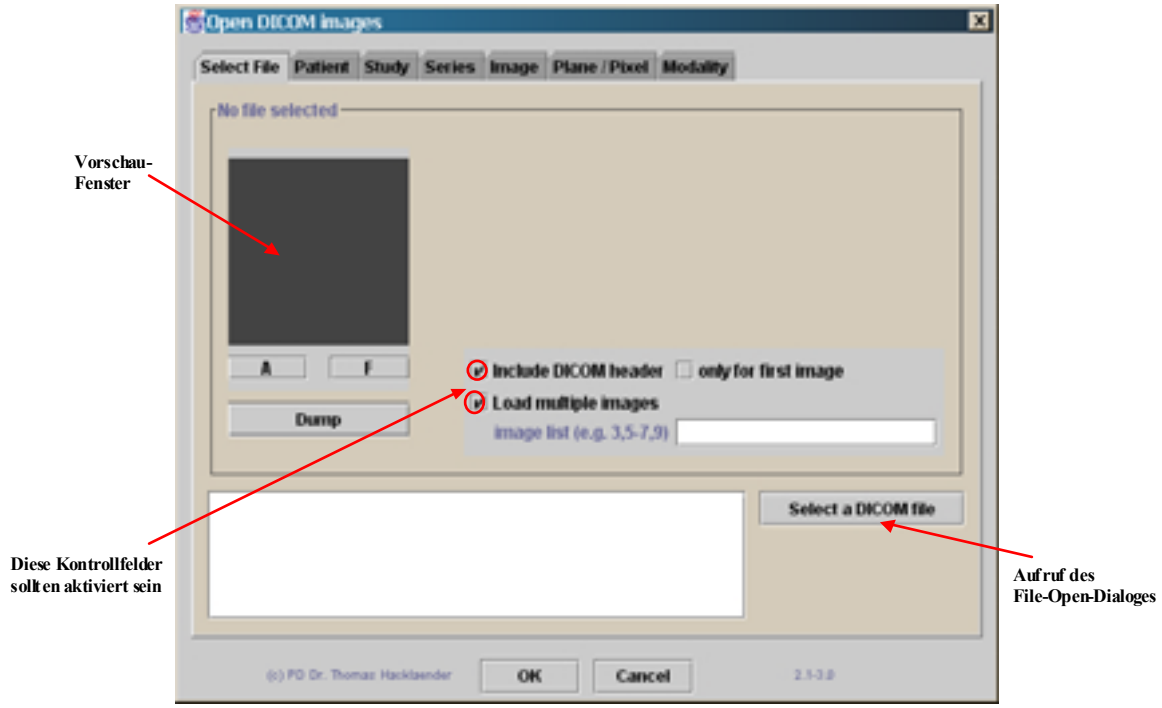


Abbildung 10.3: Dialog zum Import der Nierensequenzen.

Über die Schaltfläche *Select a DICOM file* muß nun die gewünschte Sequenz gewählt werden. Zu diesem Zweck erscheint ein Dateiauswahldialog, wie er in Abbildung 10.4 dargestellt ist.

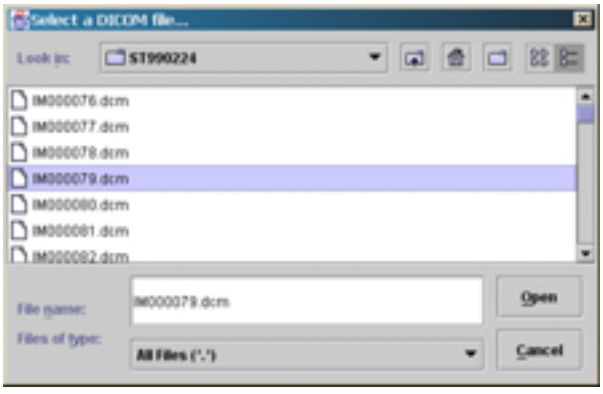


Abbildung 10.4: Ein Dateiauswahldialog (engl. File-Open-Dialog).

Die auf der CD befindlichen Beispielsequenzen im Verzeichnis *Examples* sind nach ihrer „Bearbeitungsfähigkeit“ mit dem Programm benannt. Sequenzen, bei denen die Kepstrum-Filterung gut funktioniert, wurden mit *good1* bis *good9* betitelt, Sequenzen mit akzeptablen Ergebnissen heißen *medium1* bis *medium4*. Es befinden sich auch drei Sequenzen auf der CD, bei denen die Kepstrum-Filterung oder die Klassifizierung fehlschlägt (*bad1* - *bad3*). Im jeweiligen Ordner befinden sich dann die eigentlichen Bilddaten der Nierensequenzen. Nun sollte die Auswahl eines beliebigen Bildes

erfolgen und der Dialog mit der *Open* –Schaltfläche verlassen werden. Da im Hauptfenster des Datenimports noch weitere Angaben gemacht werden müssen, ist es egal, welches Bild der Sequenz gewählt wird.

Nach Verlassen des Dateiauswahldialoges wird im Fenster des Import-Dialogs links das gewählte Bild dargestellt. In das Textfeld mit der Beschriftung *image list* muß nun die (komplette) Sequenz als numerischer Bereich (von – bis) angegeben werden. Meistens hat das erste Bild die Nummer 45; durch Unterschiede bei der Bildgewinnung ergeben sich hier aber oft unterschiedliche Startnummern. Im Listenfeld (siehe Abbildung 10.5) kann recht einfach eingesehen werden, welche Nummer das erste Bild hat. Sollte die Serie wie im Beispiel bei 40 beginnen, so ist in das oben erwähnte Feld "40-129" einzutragen[23].

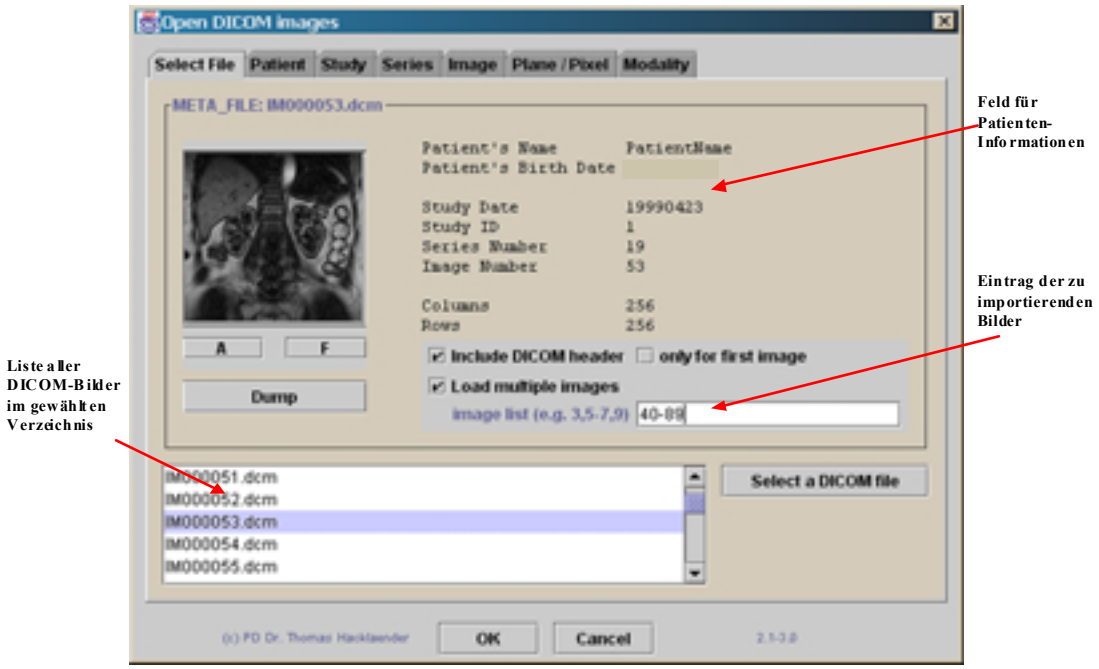


Abbildung 10.5: Dialog zum Import der Nierensequenzen.

Sobald der DICOM-Import-Dialog mit *OK* geschlossen wird, startet das Einlesen der DICOM-Bilddateien. Dabei wird der Fortschritt durch ein kleines Fenster in der Bildmitte sowie einen Balken innerhalb der Menüzeile des IMAGEJ-Hauptfenster angezeigt, siehe Abbildung 10.6.

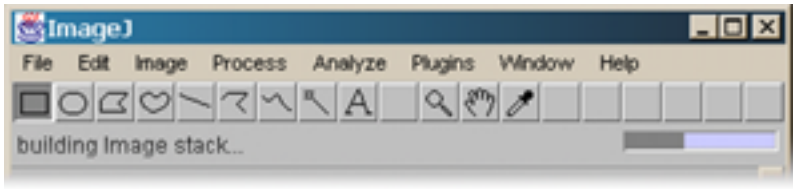


Abbildung 10.6: Fortschrittsanzeige in der Statuszeile von IMAGEJ.

Sollte der Import nicht wie beschrieben starten, so wurden möglicherweise falsche Angaben im Dialog gemacht. Leider zeigt die aktuelle Version des Import-*Plugins* dies nicht durch ein Fehlerfenster an. Vielmehr wird einfach der Dialog geschlossen, ohne daß der Bilddaten-Import gestartet wird oder ein Fortschrittsbalken angezeigt wird.

[23] Für eine korrekte Diagnosestellung ist nur in den seltensten Fällen die Erstellung der Kurven über alle 90 Bilder der Sequenz nötig. Im Beispiel in Abbildung 10.5 werden die ersten 50 Bilder importiert.

Mögliche Fehlerquellen sind:

- Der Import-Dialog wird geschlossen, ohne daß ein Bildstapel-Fenster erzeugt wird:
  - Der im Textfeld *image list* eingegebene numerische Bereich war entweder falsch oder die Eingabe wurde versehentlich vergessen.
- Es wird ein Bildstapel-Fenster erzeugt, daß nur ein Bild enthält:
  1. Das Kontrollkästchen *Load multiple images* war deaktiviert und es wurde nur das Bild importiert, welches im Listenfeld links unten markiert war, siehe Abbildung 10.5.
  2. Im Feld *image list* wurde kein Bereich (von - bis) eingegeben, sondern eine einzelne Nummer.

Wenn der Import erfolgreich abgeschlossen ist, wird die Sequenz als Bilderstapel in einem eigenen Fenster dargestellt, wie in Abbildung 10.7 gezeigt.

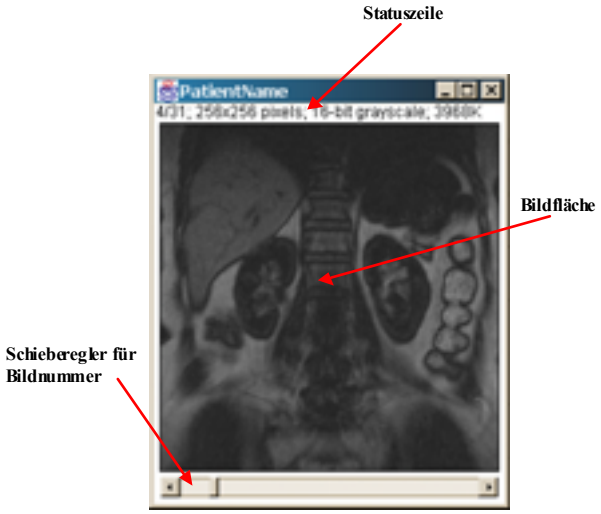


Abbildung 10.7: Darstellung der importierten Nierensequenz als Bildstapel.

10.2.3 Das *Plugin* zur renalen Funktionsanalyse

Nach dem Import der Nierensequenz kann das *Plugin* zur renalen Funktionsanalyse aufgerufen werden, indem aus dem Menü *Plugins* der Menüpunkt *RENAL FUNCTION Analysis* gewählt wird. Auch dieses *Plugin* besitzt ein eigenständiges Hauptfenster, wie in Abbildung 10.8 gezeigt.

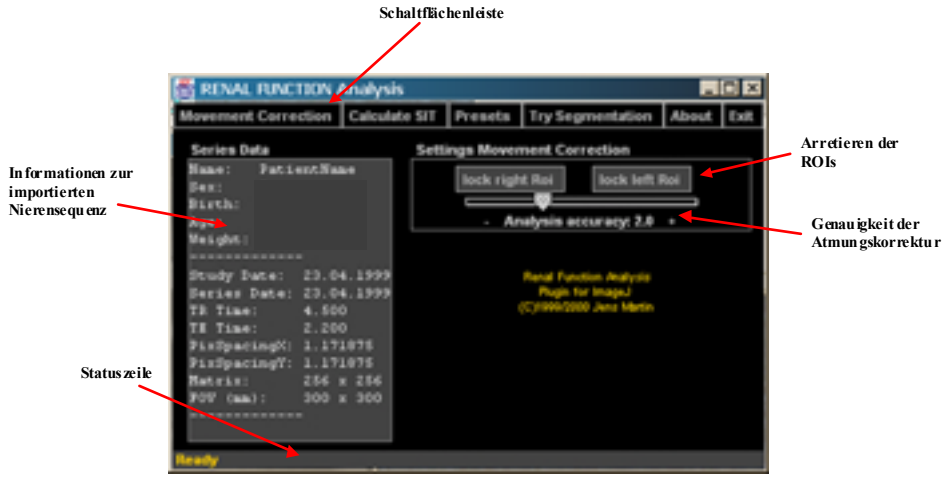


Abbildung 10.8: Hauptfenster des *Plugins* zur renalen Funktionsanalyse.

Das *Plugin* referenziert nach seinem Aufruf das Fenster der zuvor importierten Sequenz und stellt einige Erweiterungen auf dem Fester dar, die im folgenden noch genau erklärt werden. Da während der Referenzierung der Hintergrund des Bildstapel-Fensters auf schwarz umgestellt wird und einige Schaltflächen neu eingezeichnet werden, ist die erfolgreiche Durchführung des Prozesses für den Benutzer sofort sichtbar, vergleiche Abbildung 10.7 mit Abbildung 10.9.

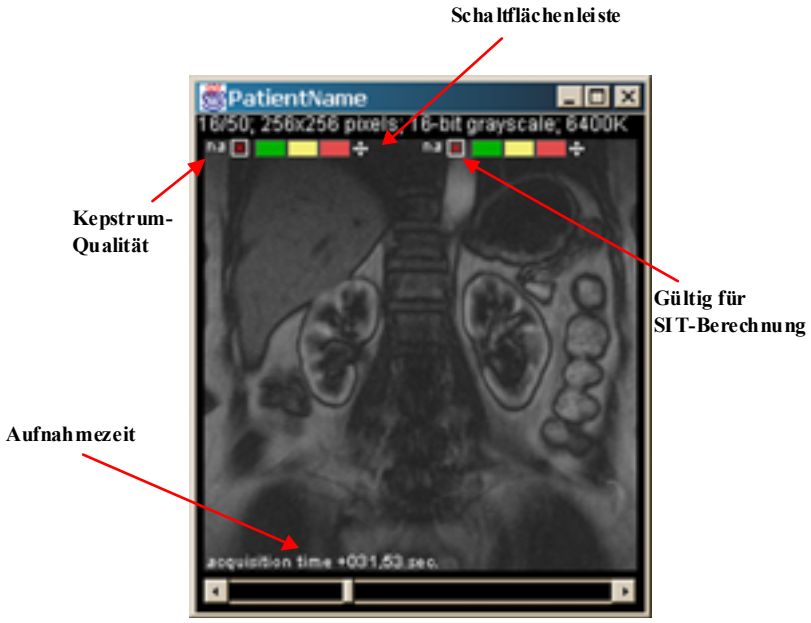
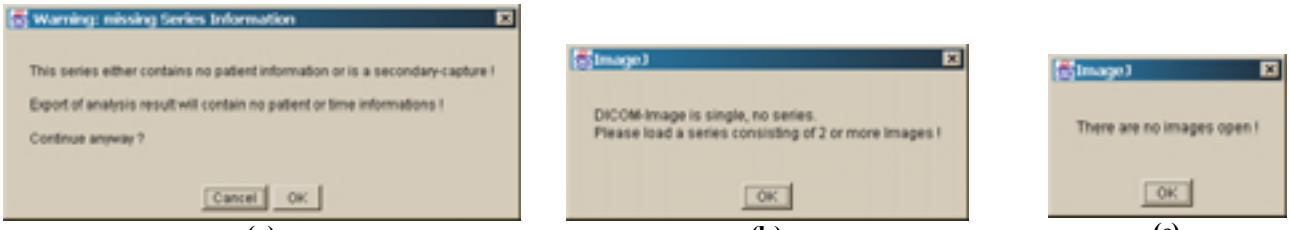


Abbildung 10.9: Fenster mit Nierensequenz Erweiterte Darstellung.

Allerdings stellt das *Plugin* zur renalen Funktionsanalyse einige Anforderungen an die importierte Sequenz. Enthält diese keine Patienten- und Aufnahmeinformationen, die zur Bearbeitung teilweise benötigt werden, so wird durch eine Warnung darauf hingewiesen, siehe Abbildung 10.10a. Die Durchführung der Funktionsanalyse kann aber trotzdem geschehen[24]. Anders sieht es aus, wenn nur ein Einzelbild statt einer Sequenz importiert wurde oder gar kein Bild geöffnet ist. Das *Plugin* startet dann erst gar nicht, sondern weist durch Fehlermeldungen auf die fehlenden Bilddaten hin, siehe Abbildung 10.10b und Abbildung 10.10c.


Warning: missing Series Information
This series either contains no patient information or is a secondary-capture !
Export of analysis result will contain no patient or time informations !
Continue anyway ?
Cancel
OK
Image 3
DICOM-Image is single, no series.
Please load a series consisting of 2 or more Images !
OK
Image 3
There are no images open !
OK

(a) (b) (c)

Abbildung 10.10: Fehlermeldungen bei fehlenden Properties oder fehlenden Bildsequenzen.

Bedienung des Bildstapel-Fensters

Das Fenster zur Anzeige der Nierensequenz stellt immer nur ein Bild der Sequenz dar. Alle anderen Bilder befinden sich im Stapel darüber oder darunter. Mit Hilfe des horizontalen Schiebereglers unterhalb der Bildfläche kann durch die Sequenz navigiert werden. Die aktuelle angezeigte Position

[24] Das Zeitintervall zwischen den Einzelaufnahmen der Sequenz wird dann auf konstant 2 Sekunden gesetzt.

(Bildnummer) wird in der Statuszeile oberhalb der Bildfläche angezeigt. Der genaue Aufnahmezeitpunkt des Bildes wird in der Statuszeile am unteren Rand des Bildes angezeigt.

Die Schaltflächenleiste am oberen Bildrand dient hauptsächlich zum Festlegen von ROIs. Jedoch zeigt sie für jede Bildhälfte eines jeden Bildes der Sequenz zwei zusätzliche Statusinformationen an. Ganz links (pro Bildhälfte) wird die zur Laufzeit berechnete Qualität der durchgeführten Kepstrum-Filterung angezeigt, siehe auch Kapitel 4.3.2. Wurde noch keine Korrektur der Atmungsbewegung durchgeführt, so ist hier der Eintrag na für nicht verfügbar (engl. not available) zu sehen. Ein "***" markiert das Bild, bei dem die Kepstrum-Filterung gestartet wurde. Die anderen möglichen Einträge werden entsprechend einer Bewertungsskala von 1 bis 5durch die Symbole "++ | + | o | – | ––" dargestellt.

Direkt rechts daneben (pro Bildhälfte) ist durch einen grünen Haken ☑ oder ein rotes Kreuz ☒ vermerkt, ob das Halbbild in die Berechnung der SIT-Kurven eingehen soll oder nicht. Entsprechend den Kepstrum-Ergebnissen werden diese Einträge automatisch gesetzt. Alle Bilder mit Ausnahme der schlechtmöglichsten Bewertung "––" werden zugelassen. Die Anzeigefläche für die Zulassung zur SIT-Berechung hat aber gleichzeitig auch die Funktion einer Schaltfläche. Der Benutzer kann durch einen einfachen Klick auf das Symbol die Auswahl umkehren und so selbst bestimmen, welche Bilder er in die SIT-Berechnung eingeben will. Durch gleichzeitiges Halten der ALT-Taste auf der Tastatur und einen Klick auf die Schaltfläche werden alle Halbbilder der Sequenz als gültig markiert. Gegenteilige Wirkung hat die ENTF-Taste. Wird diese während des Klicks gehalten, so wird die Auswahl für alle Bilder aufgehoben. Auf die Funktion der übrigen Schaltflächen der Zeile wird in Kapitel 10.2.6 und 10.2.7 noch weiter eingegangen.

10.2.4 Durchführung der Kepstrum-Filterung

Die Kepstrum-Filterung benötigt als Eingabe lediglich zwei ROIs um die Nieren herum. Diese müssen mit Hilfe des IMAGEJ-Werkzeugs zum Einzeichnen von ROIs markiert werden. Prinzipiell ist es für die Kepstrum-Filterung egal, ob rechteckige ROIs benutzt werden oder schon hier Polygon[25]-ROIs entlang der Nierenkontur markiert werden. Da im weiteren Bearbeitungsverlauf zur Berechnung der SIT-Kurven ohnehin Polygon-ROIs markiert werden müssen, wird empfohlen, dies schon zu diesem Zeitpunkt durchzuführen. Vor dem Beginn der SIT-Berechnung müssen diese dann evtl. noch verschoben werden. Sie brauchen i.d.R. aber nicht neu eingezeichnet werden.

Markieren rechteckiger ROIs

Zum Einzeichnen rechteckiger ROIs muß zunächst die IMAGEJ-Schaltfläche ▭ für rechteckige ROIs betätigt werden. Diese rastet ein und IMAGEJ zeigt in seiner Statuszeile unterhalb des Menüs die Bereitschaft an, daß nun rechteckige ROIs eingezeichnet werden können, siehe Abbildung 10.11.

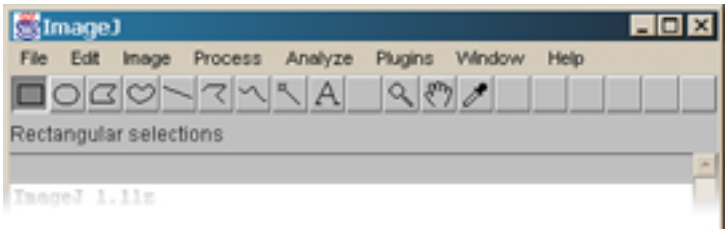


Abbildung 10.11: IMAGEJ-Schaltfläche für rechteckige ROIs.

Ziel ist es, einen rechteckigen Bereich zu markieren, der möglichst dicht die Niere umgibt, siehe Abbildung 10.12. Dazu muß mit der Maus zu einem der gedachten Eckpunkte der ROI gefahren werden. Nun wird die linke Maustaste gedrückt gehalten und die Maus zum gedachten, diagonal gegenüberliegenden Eckpunkt bewegt. IMAGEJ stellt während dieses Vorgangs die ROI schon dar.

[25] Ein geschlossener Zug aus Linien wird auch Polygon(zug) genannt.

Wird die linke Maustaste wieder losgelassen, ist die Konstruktion der ROI abgeschlossen. Die Position der ROI kann nun noch korrigiert werden, indem man mit der Maus über die ROI fährt und die linke Taste gedrückt hält. Die ROI folgt dem Mauszeiger und kann auf diese Art nachträglich positioniert werden. Falls es notwendig ist, kann auch die Größe nachträglich geändert werden. Dazu fährt man mit der Maus genau auf das kleine Quadrat im unteren rechten Eck der ROI und hält die linke Maustaste gedrückt. Nun kann die Größe durch Bewegung der Maus so lange verändert werden, bis die linke Maustaste wieder losgelassen wird. Alternativ kann aber auch einfach mit dem Einzeichnen einer neuen, rechteckigen ROI begonnen werden. IMAGEJ verwirft dann die zuvor eingezeichnete ROI, indem diese einfach gelöscht wird.

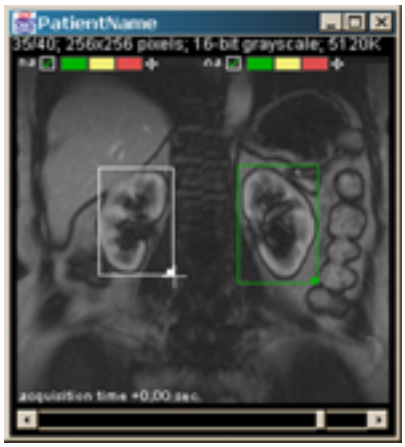


Abbildung 10.12: Markieren rechteckiger ROIs.

Markieren von Polygon-ROIs

Wird die Bildsequenz in Originalgröße dargestellt, sind die Nieren meist zu klein, um Polygon-ROIs sauber einzeichnen zu können. Es wird daher empfohlen, mittels der Vergrößerungsfunktion von IMAGEJ zunächst eine der Nieren heranzuzoomen. Dazu muß zunächst das Lupen-Symbol der Schaltflächenleiste von IMAGEJ selektiert werden. Solange die Lupen-Schaltfläche eingerastet ist, wird durch Mausklicks auf die Nierensequenz die Darstellung vergrößert oder verkleinert. Ein Klick mit der linken Maustaste vergrößert das Bild. Die Position innerhalb des Bildes, auf die dabei geklickt wurde, wird zum neuen Mittelpunkt des dargestellten (Bild-)Ausschnittes. Ein Klick mit der rechten Maustaste[26] verkleinert das Bild. Zum Markieren von Polygon-ROIs reicht normalerweise eine zweimalige Durchführung der Vergrößerung. Um dies zu erreichen, aktiviert man den Vergrößerungs-Modus von IMAGEJ und klickt mit der linken Maustaste in etwa auf den Mittelpunkt einer der Nieren, in der zweifachen Vergrößerung dann nochmals.

Für das Einzeichnen der ROIs sollte ein solches Bild der Sequenz gewählt werden, in dem die Niere bereits Kontrastmittel aufgenommen hat und die Nierenkontur gut zu erkennen ist. Mit dem Schieberegler sollte daher ein Bild oberhalb der Nummer 20 eingestellt werden.

Nun kann das Einzeichnen der Polygon-ROI beginnen. Zunächst muß dafür die Polygon-ROI-Schaltfäche im IMAGEJ-Fenster eingerastet werden. Mit der Maus sucht man sich nun eine Startpunkt für die ROI aus, der genau auf der Kontur der Niere liegt. Der (i.a. deutlich sichtbare) schwarze Rand, der die Niere gegen das umliegende Gewebe abgrenzt, sollte dabei nicht mitselektiert werden. Durch einen einfachen Klick wird der Startpunkt gesetzt; IMAGEJ zeigt dies durch einen kleinen Rahmen um den Punkt an, siehe Abbildung 10.13. Nun positioniert man die Maus über dem nächsten gewünschten Konturpunkt und legt diesen wieder durch einen einfachen Klick fest. IMAGEJ

[26] Die Maus eines Apple Macintosh hat nur eine Taste. Um einen rechten Mausklick zu erzeugen, hält man hier die -Taste während eines Mausklicks gedrückt.

zeichnet dabei ständig die ROI bis zur Position des Mauszeigers. Diese Prozedur wird wiederholt, bis die Niere einmal umrundet ist. Abgeschlossen wird der Vorgang durch ein Klick in den Kasten um den Startpunkt. Damit wird die ROI geschlossen.

Markieren von Freihand-ROIs

Der Vollständigkeit halber sei hier erwähnt, daß mittels einer Freihand-ROI ebenfalls die Kontur der Niere umfahren werden kann. Jedoch ist dies mit einer Maus als Zeigegerät auch mit viel Übung nur unpräzise möglich. Der Gebrauch eines Grafiktabletts als Eingabegeräte sollte hier wesentlich besser Ergebnisse liefern können. Leider konnten im Rahmen dieser Arbeit keine Erfahrungen über das Zusammenspiel eines Grafiktabletts mit IMAGEJ und dessen ROI-Funktionalitäten gesammelt werden. Viele professionelle Anwendungen, bei denen Bildbereiche genau markiert werden müssen, unterstützen aber Grafiktabletts als Eingabegeräte und sind durch diese meist präziser zu bedienen.

Abbildung 10.13: Markieren einer Polygon-ROI.

Start der Kepstrum-Filterung

Wie bereits erwähnt, ist als Vorbereitung für die Kepstrum-Filterung das Markieren zweier ROIs um die Nieren notwendig. Da IMAGEJ ursprünglich keine multiplen ROIs unterstützt, wurde diese Funktionalität nachträglich in das *Plugin* zur renalen Funktionsanalyse implementiert, siehe auch Kapitel 9.5.2. Das Festlegen von ROIs geschieht mittels der *lock left*- und *lock right*-Schaltflächen auf dem Hauptfenster des *Plugins*. Aktive ROIs sind weiß und ändern ihre Farbe beim Arretieren nach Grün. Um Mißverständnissen vorzubeugen, sei erwähnt, daß sich die rechte Niere – und damit auch die rechte ROI – auf der linken Bildhälfte befindet und umgekehrt. Dies rührt daher, daß die Positionen der Nieren aus Sicht des Patienten beschrieben werden. Da die Sequenz in etwa eine Frontalaufnahme des Bauchraums darstellt, sind rechts und links vertauscht. Diese seitenverkehrte Beschriftung findet sich auch bei einigen der im folgenden besprochenen Dialoge, Menüs oder Schaltflächen.

Bevor die Kepstrum-Filterung durch einen Klick auf die Schaltfläche *Movement Correction* im Menü des *Plugin*-Hauptfensters gestartet wird, sollte noch die Genauigkeit der Berechnung eingestellt werden. Der Schieberegler im rechten Teil des Hauptdialoges ist dafür verantwortlich. Die Genauigkeit m läßt sich im Bereich zwischen 1,0 und 4,0 einstellen. Dieser Wert entspricht dem in Kapitel 7.6 besprochenen Skalierungsfaktor m, der vor der Berechnung auf die Bilder angewendet wird. Die detektierte Pixelverschiebung ist $1 / m$. Bei einer Genauigkeit von beispielsweise 2,5 wird die Verschiebung zwischen Referenz- und Folgebild auf $1 / 2{,}5 = 0{,}4$ Pixel genau bestimmt.

Während die Kepstrum-Filterung läuft, wird sowohl das Fenster der Nierensequenz als auch das Hauptfenster des *Plugins* für Eingaben von Tastatur und Maus gesperrt. Auf dem Hauptfenster wird eine Schaltfläche *Cancel* eingeblendet, der die Kepstrum-Filterung abbrechen kann. Der Fortschritt der Filterung wird wie schon beim Import der Bilddaten durch einen Balken im IMAGEJ-Fenster angezeigt, siehe Abbildung 10.14. Zusätzlich wird die Statuszeile von IMAGEJ benutzt, um auch textuelle Informationen über den Fortschritt der Berechnungen zu vermitteln.

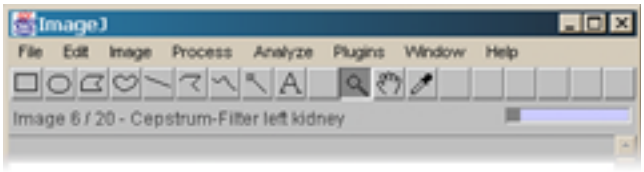
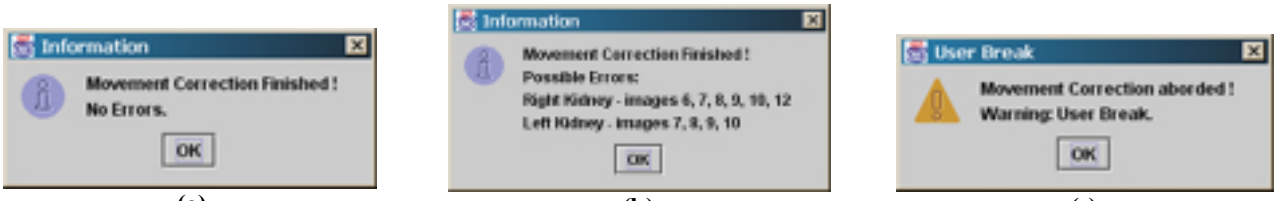


Abbildung 10.14: Anzeige des Fortschritts der Kepstrum-Filterung im IMAGEJ-Fenster.

Die Beendigung der Kepstrum-Filterung wird dem Benutzer durch ein kleines Informationsfenster angezeigt. Bei schlechten Ergebnissen werden zusätzlich die Bildnummern der Sequenz angegeben, deren Atmungskorrektur wahrscheinlich fehlgeschlagen ist. Für eine spätere, manuelle Nachjustierung einzelner Bildhälften (vergleiche Kapitel 10.2.5) kann es sinnvoll sein, die Nummern zu notieren.



(a) (b) (c)

Abbildung 10.15: Meldungen nach Beendigung der Kepstrum-Filterung.

10.2.5 Berechnung der SIT-Kurven

Visuelle Kontrolle der Kepstrum-Ergebnisse

Die Berechnung der SIT-Kurven setzt eine erfolgreiche Korrektur der Atmungsbewegung voraus. Die berechneten, qualitativen Bewertungen finden sich in den schon beschriebenen Statusfeldern am oberen Rand der Bildfläche. Eine visuelle Kontrolle kann schnell durchgeführt werden, indem die Sequenz mit dem Schieberegler „durchgeblättert“ wird und die Deckungsgleichheit der Nieren für alle Bilder geprüft wird. Dafür kann es sinnvoll sein, vorher mit der Lupenfunktion auf zwei- oder vierfache Vergrößerung umzuschalten.

Vorbereitung der SIT-Berechung

Wie die im Kapitel zuvor besprochene Kepstrum-Filterung, so setzt auch die Berechnung der SIT-Kurven das Markieren zweier ROIs voraus. Für eine exakte Berechnung werden hier aber zwingend Polygon-ROIs benötigt. Sollten diese bereits zur Kepstrum-Filterung eingezeichnet worden sein, ist es eventuell erforderlich, durch kleine Verschiebungen eine Deckung mit der Kontur der Niere wiederherzustellen. Durch einen Klick auf die entsprechende orange-farbige *unlock*-Schaltfläche auf dem *Plugin*-Hauptfenster wird die Arretierung der ROI aufgehoben und die Farbe wechselt von grün nach weiß. Die ROI kann nun verschoben werden, indem mit der Maus darauf gezeigt wird und die linke Maustaste gedrückt gehalten wird. Zum leichteren Positionieren empfiehlt es sich wieder, auf ein Bild aus dem hinteren Bereich der Sequenz zurückzugreifen. Durch abermalige Betätigung der *lock*-Schaltfläche muß die ROI wieder arretiert werden.

Für den Fall, daß zur Durchführung der Kepstrum-Filterung rechteckige ROIs markiert wurden, müssen diese gelöscht und durch Polygon-ROIs ersetzt werden. Indem man für eine bestehende ROI die Arretierung aufhebt und einfach mit dem Einzeichnen einer neuen ROI beginnt, wird die alte automatisch von IMAGEJ verworfen. Alternativ kann auch die ENTF-Taste gedrückt gehalten werden und durch einen Klick auf die der Bildhälfte entsprechende grüne Schaltfläche im Bildfenster die ROI gelöscht werden. Für das erforderliche Einzeichnen einer Polygon-ROI wird auf den entsprechenden Abschnitt im vorherigen Kapitel 10.2.4 verwiesen.

Korrektur der Verschiebung von Einzelbildern

In Ausnahmefällen kann es vorkommen, daß einzelne Halbbilder bei der Kepstrum-Filterung nicht richtig korrigiert werden, siehe Kapitel 7.6 und 7.7.2. Der Algorithmus zur Bearbeitung kann dies i.d.R. durch die numerische, qualitative Kontrolle (siehe Kapitel 4.3.2) feststellen und weist im Meldungsfenster nach Beendigung der Kepstrum-Filterung darauf hin, siehe Abbildung 10.15b.

Eine manuelle Nachjustierung einzelner Halbbilder ist aber selbstverständlich möglich. Dazu ist es am sinnvollsten, zunächst – wie im Abschnitt zuvor beschrieben – die ROIs anhand eines Bildes der Sequenz einzuzeichnen, für welches die Kepstrum-Filterung erfolgreich war. Dann fährt man mit dem Schieberegler zu dem nachbearbeitungs-bedürftigen Bild der Sequenz und startet das Justierungs-Modul für die gewünschte Bildhälfte durch einen Klick auf die -Schaltfläche im Bildfenster. Es erscheint ein kleiner Dialog neben dem Bildfenster, wie in Abbildung 10.16 dargestellt.

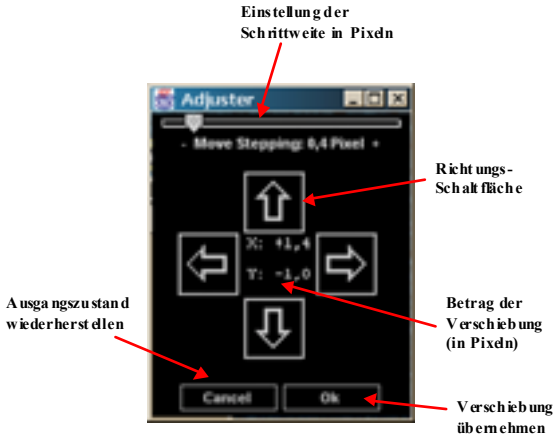


Abbildung 10.16: Hilfsdialog zur manuellen Justierung der Verschiebung.

Die Bildhälfte läßt sich durch Mausklicks auf die vier Richtungsschaltflächen verschieben. Die Schrittweite der Verschiebung (pro Mausklick) in Pixeln wird durch den Schieberegler am oberen Rand des Dialoges eingestellt. Der Benutzer kann dabei einen Wert im Bereich von 0,1 bis 2,0 Pixeln wählen. Der Betrag der bereits durchgeführten Verschiebung seit Öffnen des Dialoges wird im Textfeld in der Mitte der Richtungsschaltflächen angezeigt. Die Verschiebung wird durch einen schnellen Interpolations-Algorithmus (siehe Kapitel 5.4.2) durchgeführt und ist nach einer kurzen Berechnungszeit sichtbar. Je nach Geschwindigkeit des Rechners[27] ergibt sich sogar ein Echtzeit-Algorithmus, bei dem die Zeit für die Berechnung kaum noch wahrnehmbar ist.

Während der Verschiebung des Bildes behalten alle eingezeichneten ROIs ihre ursprüngliche Position und werden nicht mit verschoben. Dadurch ist es einfach möglich, die manuelle Justierung so durchzuführen, daß die Niere optimal unter die (auf alle Bilder der Sequenz projizierten) ROI geschoben wird. Durch einen Klick auf *OK* wird die Verschiebung beibehalten; *Cancel* stellt den Ausgangszustand vor Aufruf des Verschiebungs-Moduls wieder her.

[27] Ein PENTIUM III mit mindestens 500 MHz wird empfohlen

Das manuell justierte Halbbild sollte schließlich noch durch einen Klick auf die -Schaltfläche für die Berechnung der SIT-Kurven zugelassen werden.

Start der SIT-Berechnung

Durch einen Mausklick auf die *Calculate SIT*-Schaltfläche im Hauptfenster des *Plugins* wird die Berechnung gestartet. Sie dauert auch auf langsamen (PENTIUM-) Rechnern für eine komplette Sequenz nur einige Sekunden. Wie bei der Kepstrum-Filterung, so werden auch hier wieder alle Schaltflächen des Hauptfensters gesperrt und die *Cancel*-Schaltfläche aktiviert. Wenn nicht anders durch die Voreinstellungen (vergleiche Kapitel 10.2.8) festgelegt, werden berechnete Zwischenergebnisse in textueller Form im IMAGEJ-Fenster ausgegeben.

Das Ergebnisfenster

Die Ergebnisse werden in einem eigenen Fenster mit drei Koordinatensystemen dargestellt, wie in Abbildung 10.17 gezeigt. Der linke Rand des Fensters zeigt die Pixel-Klassifizierung in zweifacher Vergrößerung als eingefärbte Abbildungen. Pixel der Nierenrinde sind gelb markiert, Pixel des Nierenmarks rot. Alle anderen Pixel behalten ihren ursprünglichen Grauton. Durch einen Doppelklick auf eines der Klassifizierungsbilder wird ein neues IMAGEJ-Bildfenster erzeugt, welches das eingefärbte Bild enthält, siehe ebenfalls Abbildung 10.17. Dieses kann z.B. weiter vergrößert werden, um die Ergebnisse genauer zu überprüfen.

Die gleiche Farbkodierung wurde auch für die Kurvendarstellung im mittleren Bereich des Fensters übernommen. Außer der Anzeige der SIT-Kurven besitzen die Koordinatensysteme keine weitere Funktionalität. Lediglich die aktuelle Mausposition wird noch verarbeitet. Fährt der Benutzer mit der Maus auf das Koordinatensystem (und auf einen Punkt der Kurve), so wird die dazugehörige Aufnahmezeit sowie der Grauwert in einem kleinen Index angezeigt, siehe Abbildung 10.18a.

Im rechten Bereich des Fensters finden sich für jedes Kurvenpaar drei Kontrollkästchen, mit deren Hilfe Darstellungsoptionen der Kurve an- bzw. ausgeschaltet werden können. Die Darstellung einer Kurve mit Anzeige der Standardabweichungen wird in Abbildung 10.18b gezeigt. Durch die drei Schaltflächen im unteren Bereich können der Export sowie die im Kapitel 10.2.6 beschriebene Justierung der Klassifizierungsergebnisse aufgerufen werden. *OK* schließt das Ergebnisfenster.

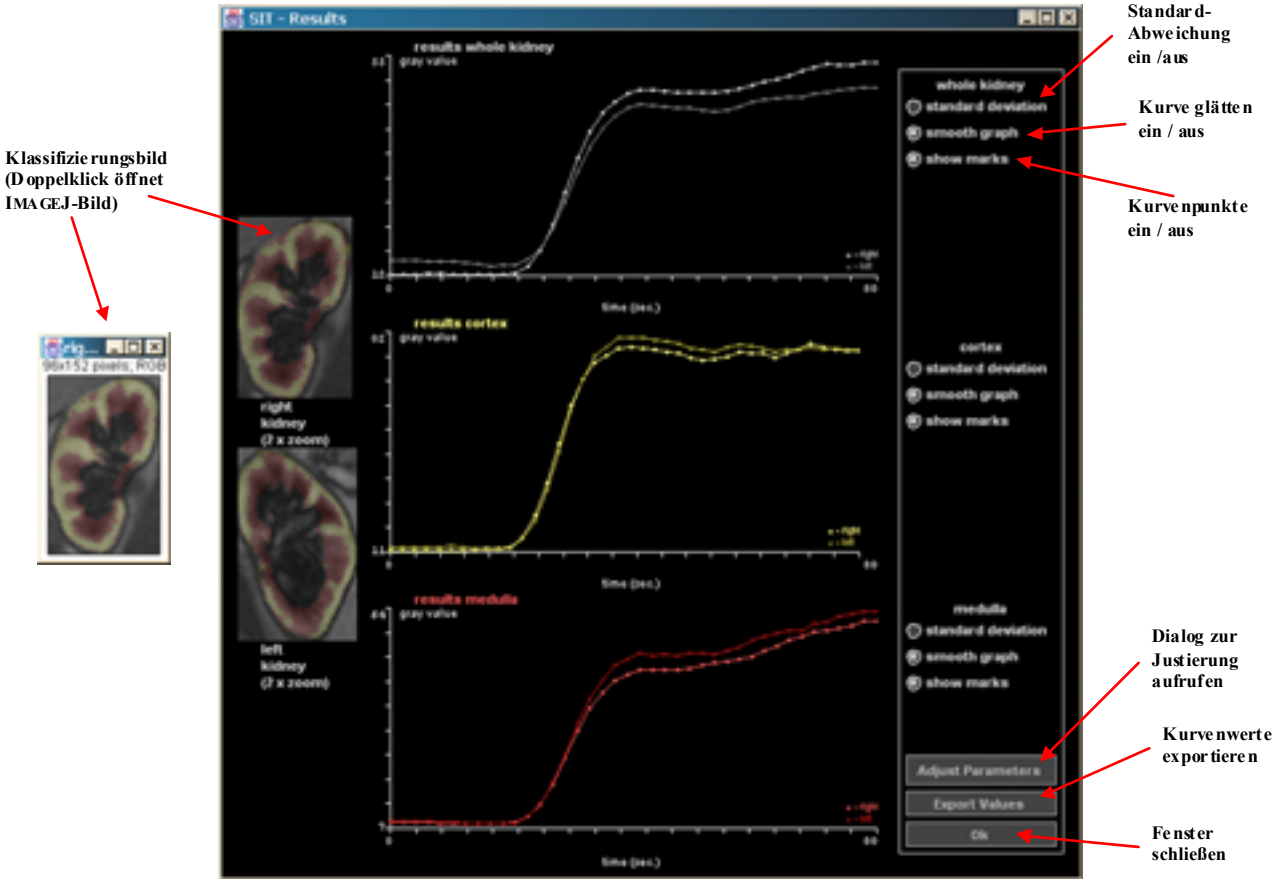


Abbildung 10.17: Ergebnisfenster der SIT-Kurven-Berechnung.

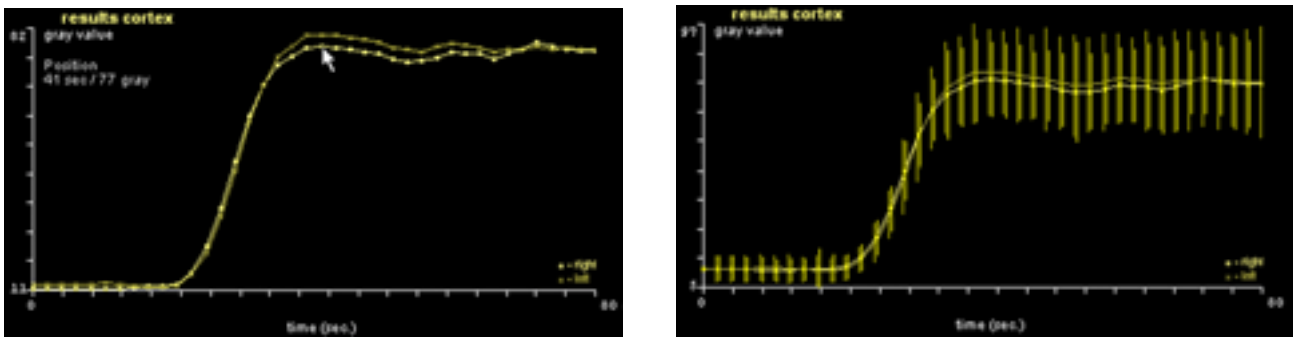


Abbildung 10.18: Berechnete Werte an der Mausposition (a), Darstellung der Standardabweichung (b).

Export der Ergebnisse

Beim Klick auf *Export* wird der schon in Kapitel 10.2.2 vorgestellte Dateiauswahldialog aufgerufen. Der Benutzer kann hier Zielverzeichnis und Dateinamen für die exportierte Textdatei wählen. Durch Betätigung der Schaltfläche *Save* wird die Datei geschrieben.

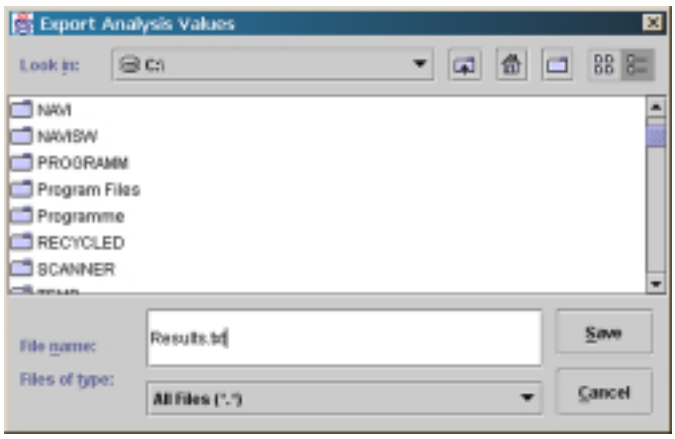


Abbildung 10.19: Dialog zum Export der berechneten Kurvenwerte als Textdatei.

10.2.6 Justierung der Klassifizierungsergebnisse

Wird in dem im vorherigen Kapitel 10.2.5 beschriebenen Ergebnisfenster die Schaltfläche *Adjust Results* angeklickt, so öffnet sich ein weiterer Dialog, wie in Abbildung 10.20 dargestellt. Hier bietet sich dem Benutzer die Möglichkeit, die vom Programm (automatisch) berechneten Pixel-Klassifizierungen durch eine Veränderung der Schwellenwerte zu justieren, vergleiche Kapitel 8.1.

Im linken Bereich des Fensters werden nochmals die eingefärbten Klassifizierungsbilder für die rechte und linke Niere dargestellt. Direkt links daneben werden zusätzlich Bildausschnitte der Nieren aus dem Bild der Sequenz dargestellt, das dem berechneten Wendepunkt entspricht, vergleiche Kapitel 8.1.2. Durch Bewegen der Schieberegler kann der Benutzer nun die Schwellenwerte verändern. Dank des Echtzeit-Algorithmus können die neuen Pixel-Klassifizierungen sofort berechnet und angezeigt werden. Sinn der Justierung ist es, die Schwellenwerte derart zu verändern, daß die Klassifizierungsergebnisse im eingefärbten Bild optimal mit dem links daneben dargestellten Wendepunktbild übereinstimmen.

Durch einen Klick auf die Schaltfläche *Apply* werden die neuen Schwellenwerte endgültig übernommen und die SIT-Kurven neu berechnet. *Cancel* schließt den Dialog und verwirft dabei die veränderten Werte.

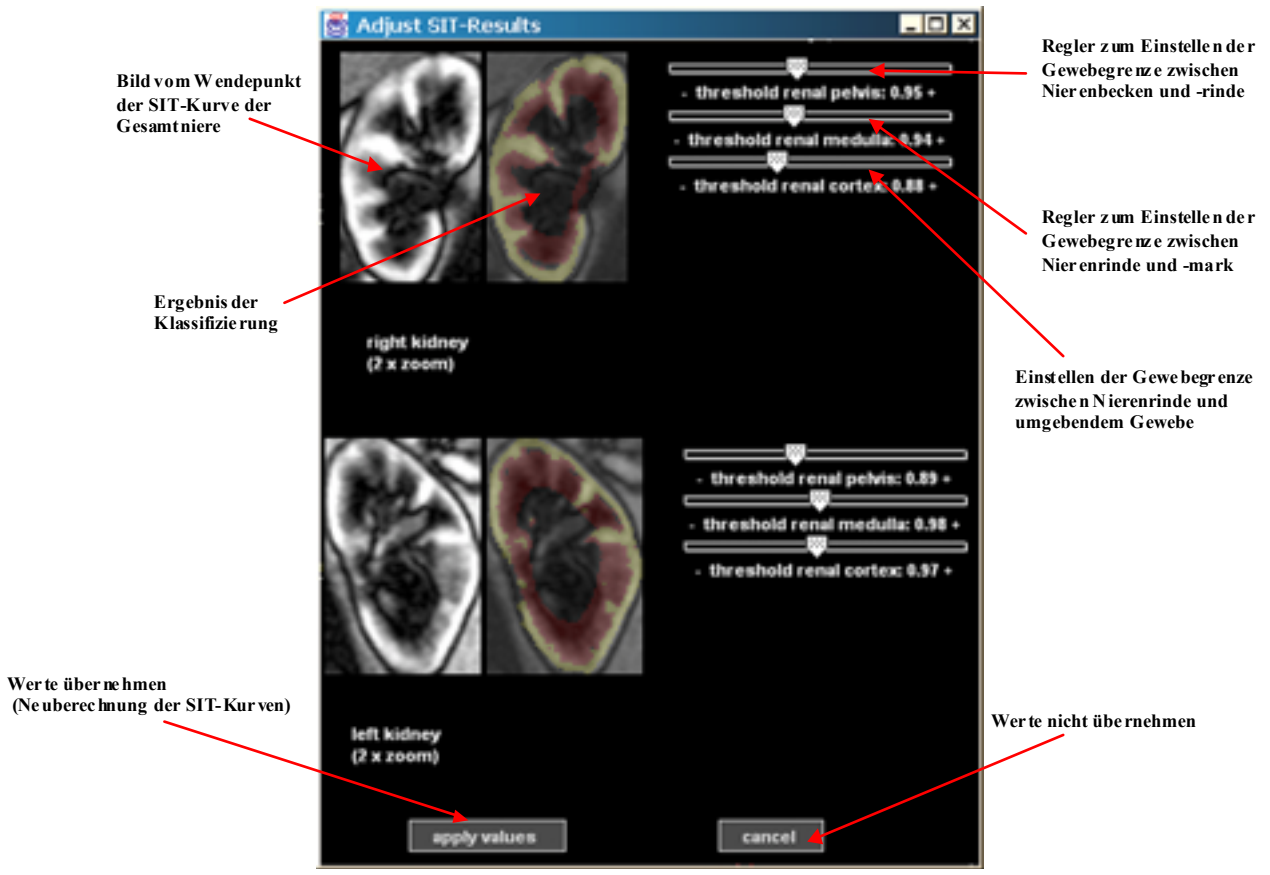

Adjust SIT-Results
Bild vom Wendepunkt der SIT-Kurve der Gesamtniere
Regler zum Einstellen der Gewebegrenze zwischen Nierenbecken und -rinde
threshold renal pelvis: 0.95
threshold renal medulla: 0.94
threshold renal cortex: 0.88
Regler zum Einstellen der Gewebegrenze zwischen Nierenrinde und -mark
Ergebnis der Klassifizierung
right kidney (2 x zoom)
Einstellen der Gewebegrenze zwischen Nierenrinde und umgebendem Gewebe
threshold renal pelvis: 0.89
threshold renal medulla: 0.98
threshold renal cortex: 0.97
Werte übernehmen (Neuberechnung der SIT-Kurven)
left kidney (2 x zoom)
Werte nicht übernehmen
apply values
cancel

Abbildung 10.20: Dialog zur Justierung der Klassifizierungsergebnisse.

10.2.7 Manuelle Bearbeitung von Nierensequenzen

Eine manuelle Bearbeitung wird immer dann notwendig, wenn entweder die Kepstrum-Filterung sehr schlechte Ergebnisse liefert oder die Gewebedifferenzierung auch nach einer Justierung nicht den Anforderungen genügt. Der erste Fall hängt meistens direkt von der Aufnahmequalität ab, siehe auch Kapitel 7.7 und 7.8. Eine fehlerhafte Klassifizierung kann bei größeren Gewebedefekten auftreten, die vom Algorithmus nicht berücksichtigt sind, z.B. Nierentumoren oder Niereninfarkten.

In beiden Fällen ist es erforderlich, die Auswertung im herkömmlichen Stil durch das Einzeichnen mehrerer ROIs pro Halbbild durchzuführen. Wie bereits in der Einleitung zur Arbeit gesagt, wurde diese Funktionalität mit in das Programm übernommen, um einen kompletten Arbeitsplatz zur Nierenperfusionsuntersuchung bieten zu können.

Einzeichnen mehrerer ROIs

Im folgenden werden wieder die schon beschriebenen Polygon- oder Freihand-ROIs benötigt, siehe Kapitel 10.2.4. Für das Arretieren der ROIs wird die Schaltflächenleiste am oberen Rand des Bildes benötigt, siehe Abbildung 10.22. Wird beabsichtigt, eine ROI innerhalb der Nierenrinde zu markieren, so kann diese zunächst wie gewohnt eingezeichnet werden (eventuell unter geeigneter Vergrößerung des Bildausschnittes). Durch einen Klick auf die gelbe Schaltfläche wird die fertige ROI arretiert. Wird beim Klick auf eine farbige Schaltfläche einer Bildhälfte keine weitere Taste der Tastatur gedrückt gehalten und wird die Schaltfläche zum ersten mal betätigt, so erscheint ein Dialog mit einer Abfrage, siehe Abbildung 10.21. Hier muß der Benutzer entscheiden, ob er die gerade eingezeichnete ROI auf alle Bilder der Sequenz projizieren und arretieren will oder ob die ROI nur im gerade angezeigten Bild festgesetzt werden soll.

Indem das Kontrollkästchen *don't show this dialog again* im Dialog aktiviert wird, wird das weitere Erscheinen der Nachfrage bei einem Klick unterbunden. Das Programm geht dann automatisch davon aus, daß die ROI nur im aktuellen Bild arretiert werden soll.

Abbildung 10.21: Abfrage beim Festsetzen von gewebespezifischen ROIs.

Der Abfragedialog erklärt dem Benutzer gleichzeitig auch die Funktion der beiden Tasten ALT und ENTF. Wird während des Klicks auf eine Schaltfläche die ALT-Taste gehalten, so wird die ROI projiziert und für alle Bilder arretiert.

Das Einzeichnen von ROIs für Geweberegionen des Nierenmarks geschieht auf die gleiche Art und Weise. Zum Festlegen von Mark-ROIs muß die rote Schaltfläche benutzt werden. Pro Halbbild können vom Benutzer jeweils bis zu 100 ROIs für Mark und Rinde eingezeichnet werden; diese Obergrenze sollte für alle Anwendungsfälle reichen.

Abbildung 10.22: Einzeichnen mehrerer gewebespezifischer ROIs (gelb, rot), Gesamt-ROI (grün).

Das Festlegen von ROIs für die Gesamtnieren kann entweder durch den schon beschriebenen Weg über die *lock*-Schaltfächen auf dem Hauptfenster geschehen oder durch einen Mausklick auf die grüne Schaltfläche. Hier kann der Benutzer dann wieder wählen, ob die ROI projiziert werden soll oder

nicht. Werden die Schaltflächen des Hauptfensters benutzt, findet immer eine Projektion statt. Pro Halbbild kann jeweils nur eine ROI für das Gesamtorgan festgelegt werden.

Verschieben von ROIs

ROIs können nur im aktiven Zustand verschoben werden. Sind sie einmal arretiert, können sie nicht mehr durch ein Festhalten mit der Maus bewegt werden. Indem mit der linken Maustaste ein Doppelklick in den (gedachten) Mittelpunkt einer festgesetzten ROI durchgeführt wird, kann diese aktiviert werden und ändert ihre Farbe nach weiß. Die ROI kann nun z.B. neu positioniert werden oder – im Falle von rechteckigen ROIs – in ihrer Größe verändert werden.

Für eine Sequenz, bei der die Kepstrum-Filterung versagt, können so für ein Bild zunächst alle ROIs eingezeichnet und projiziert werden. Dann wählt man das nächste Bild und paßt die Verschiebung aller ROIs der Niere an. Wird dies für alle Bilder der Sequenz wiederholt, so entspricht das im wesentlichen dem ursprünglichen Bearbeitungsweg.

Löschen von ROIs

Indem die ENTF-Taste gedrückt gehalten wird, während der Benutzer auf eine der ROI-Schaltflächen klickt, können alle ROIs dieses Typs gelöscht werden. Soll nur eine einzelne ROI entfernt werden, so aktiviert man sie durch einen Doppelklick, wählt den Rechteck-ROI-Modus von IMAGEJ und führt einen einfachen Klick in einem Bereich des Bildes ohne ROIs aus.

Berechnung der SIT-Kurven

Sind alle ROIs eingezeichnet, kann durch einen Mausklick auf die Schaltfläche *Calculate SIT* im Hauptfenster die Kurvenberechnung gestartet werden. Das Programm erkennt automatisch die multiplen ROIs und wertet nur diese aus. Die automatische Gewebeklassifizierung wird nicht durchgeführt. Daher kann es sein, daß im Ergebnisfenster keine Kurve für einen Gewebetyp angezeigt wird, wenn für diesen keine spezifischen ROIs markiert worden sind.

10.2.8 Voreinstellungen

Der Dialog zur Steuerung der Voreinstellungen wird durch die Schaltfläche *Presets* im Hauptfenster aufgerufen. Der Dialog enthält drei Karteikarten, die jeweils unterschiedliche Voreinstellungen steuern. Durch einen Klick auf die Lasche am oberen Rand kann die gewünschte Karte aktiviert werden. Durch einen Klick auf die Schaltfläche *Default Values* werden die aktuellen Einstellungen durch fest im Programm gespeicherte Werte überschrieben. *Cancel* schließt den Dialog, ohne die veränderten Werte abzuspeichern, und *OK* übernimmt die veränderten Werte.

Die Karteikarte *Cepstrum Settings*

Diese Karteikarte steuert die Voreinstellungen für die Kepstrum-Filterung, siehe Abbildung 10.23. Im Anschluß an die Abbildung wird jeder Punkt tabellarisch erklärt.

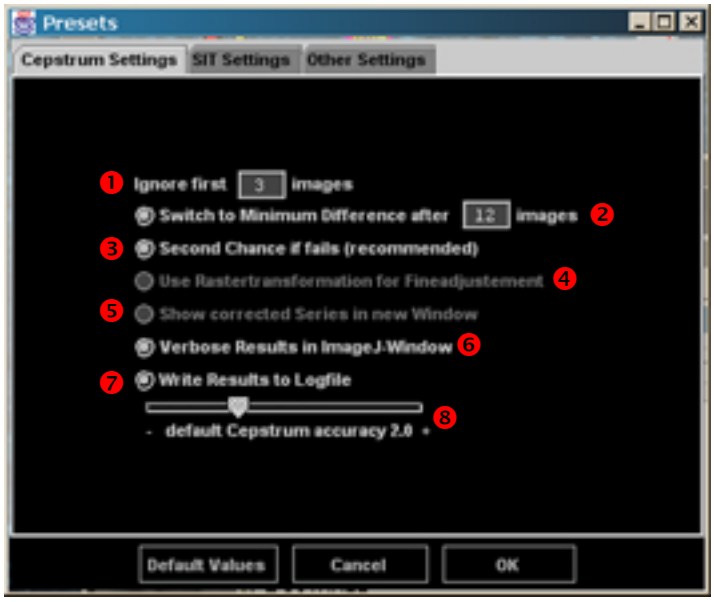


Abbildung 10.23: Die Karteikarte *Cepstrum Settings*.

Nummer	Beschreibung / Funktion
❶	Legt fest, ab welcher Bildnummer die importierte Sequenz bearbeitet werden soll. Der Magnetresonanztomograph benötigt eine kurze „Einschwingphase“, innerhalb der z.B. Grauwerte und Helligkeit der Bilder schwanken können, siehe Kapitel 3.2. Voreinstellung: 3
❷	Hier kann festgelegt werden, ob nach einer gewissen Anzahl von Durchläufen der Algorithmus zur Atmungs-Korrektur anstatt der Kepstrum-Filterung eine minimale Differenz-Filterung benutzen soll. Ist dieser Punkt aktiviert, so wird während der Kepstrum-Filterung für die rechte und linke Niere die Verschiebungsachse protokolliert. Hat sich diese für die angegebene Anzahl Durchläufe *nicht* mehr verändert, so wird angenommen, daß alle Organverschiebungen auf dieser Achse liegen. Dann kann ein minimales Differenzverfahren eingesetzt werden, das schneller berechnet werden kann als die Kepstrum-Filterung. Voreinstellung: an / 12
❸	Hier kann eingestellt werden, ob der Algorithmus zur Atmungs-Korrektur (siehe auch Abbildung 4.5) fehlertolerant arbeiten soll. Es wird dringend empfohlen, diesen Punkt zu aktivieren. Voreinstellung: an
❹	Bestimmt, ob die Rastertransformation zur zusätzlichen Anpassung von qualitativ schlecht zu filternden Sequenzen benutzt werden soll, vergleiche Kapitel 7.7.2. Dieser Punkt ist deaktiviert, da diese Option nur theoretisch diskutiert wurde, aber aus Zeitgründen nicht umgesetzt werden konnte. Voreinstellung: aus
❺	Bestimmt, ob die durch die Kepstrum-Filterung korrigierte Serie in einem neuen Fenster angezeigt werden soll. Voreinstellung: aus

❻	Hier kann angegeben werden, ob Zwischenergebnisse der Kepstrum-Filterung textuell im IMAGEJ-Hauptfenster ausgegeben werden sollen. Voreinstellung: an
❼	Wenn dieser Punkt aktiviert ist, werden Zwischenergebnisse der Kepstrum-Filterung in einem Log-File mitgeschrieben, siehe Kapitel 9.6.2. Voreinstellung: an
❽	Bestimmt die Genauigkeit der Kepstrum-Filterung, die beim ersten Öffnen des Hauptfensters der renalen Funktionsanalyse eingestellt ist (siehe Kapitel 10.2.3 und 7.6.). Voreinstellung: 2,0

Tabelle 10.1: Beschreibung der auf der Karteikarte *Cepstrum Settings* wählbaren Voreinstellungen.

Die Karteikarte *SIT Settings*

Hier kann der Benutzer Einstellungen vornehmen, die sich auf die Berechnung und Darstellung der Signal-Intensitäts-Zeit-Kurven auswirken. Abbildung 10.24 zeigt die Karte. Im Anschluß an die Abbildung werden die Einstellungen wieder tabellarisch erklärt.

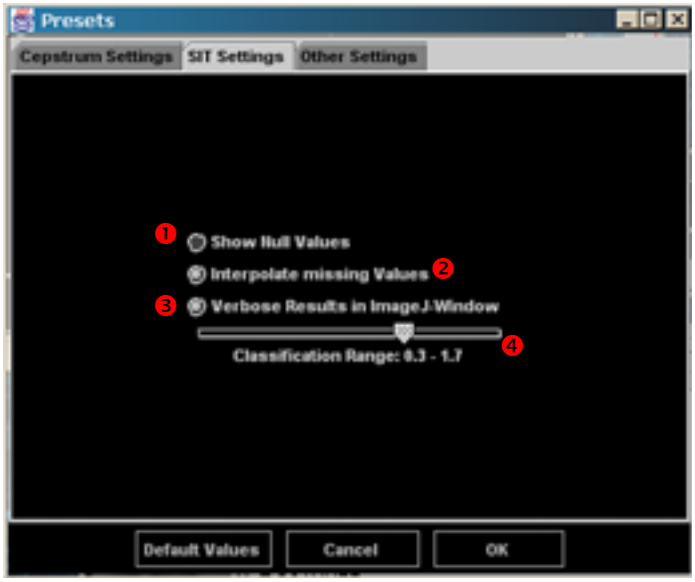


Abbildung 10.24: Die Karteikarte *SIT Settings*.

Nummer	Beschreibung / Funktion
❶	Legt fest, ob Nullwerte bei der Darstellung der SIT-Kurven angezeigt werden sollen. Wenn dieses Kontrollkästchen aktiv ist, sieht der Kurvenverlauf evtl. sehr unschön aus, falls sich bei einzelnen Bildern keine Berechnungswerte ergeben (dies kann insbesondere bei der manuellen Bearbeitung passieren, siehe Kapitel 10.2.7) Voreinstellung: aus
❷	Legt fest, ob fehlende Kurvenwerte durch Interpolation aus den Nachbarpositionen errechnet werden sollen. Wenn dieses Kontrollkästchen nicht aktiv ist, entstehen unter Umständen Lücken im Kurvenverlauf. Voreinstellung: an

❸	Steuert die textuelle Ausgabe von Zwischenergebnissen der SIT-Berechung im IMAGEJ-Fenster. Voreinstellung: an
❹	Legt den Bereich fest, innerhalb dessen die Schwellenwerte zur Nachjustierung verändert werden können, siehe Abbildung 10.20 und Kapitel 10.2.6. Voreinstellung: 0,5-1,5

Tabelle 10.2: Beschreibung der auf der Karteikarte *SIT Settings* wählbaren Voreinstellungen.

Die Karteikarte *Other Settings*

Hier kann der Benutzer Einstellungen vornehmen, die sich auf die Berechnung und Darstellung der Signal-Intensitäts-Zeit-Kurven auswirken. Abbildung 10.25 zeigt die Karte. Im Anschluß an die Abbildung werden die Einstellungen wieder tabellarisch erklärt.

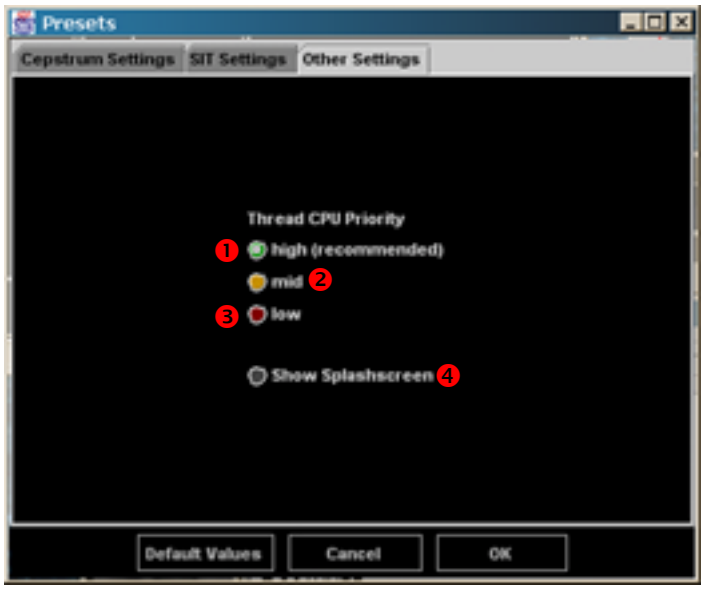


Abbildung 10.25: Die Karteikarte *Other Settings*.

Nummer	Beschreibung / Funktion
❶-❸	Legt fest, in welchem Maß die Rechenzeit des Prozessors für die Kepstrum- und SIT-Berechnungen genutzt wird. Bei der Einstellung *high* werden die Berechnungen am schnellsten durchgeführt, jedoch laufen alle anderen aktiven Programme sehr langsam. Voreinstellung: high
❹	Legt fest, ob das Startbild (siehe Abbildung 10.1) beim Aufruf des *Plugins* zur renalen Funktionsanalyse eingeblendet werden soll. Voreinstellung: an

Tabelle 10.3: Beschreibung der auf der Karteikarte *Other Settings* wählbaren Voreinstellungen.

10.2.9 Sonstige Bedienungselemente

Automatische Detektion der Organkontur

Wie bereits in Kapitel 7.8 beschrieben, stellt die automatische Detektion der Organkontur ein Problem dar, dessen Lösung äußerst komplex ist. Die in Kapitel 9.10.1 beschriebene Umsetzung wurde aber nicht verworfen, sondern als eigenständiger Menüpunkt mit in das *Plugin*-Hauptfenster übernommen. Um eine Detektion durchzuführen, muß zunächst eine rechteckige ROI um eine der Nieren gezeichnet werden. Diese darf dann – im Gegensatz zum sonstigen Vorgehen – nicht arretiert werden. Die automatische Detektion wird durch einen Mausklick auf die Schaltfläche *Try Segmentation* gestartet. Die Berechnung dauert auch auf langsamen Systemen nie länger als etwa 2 Sekunden. Die rechteckige ROI wird dann automatisch von IMAGEJ durch eine Polygon-ROI ersetzt, die das Ergebnis der Detektion darstellt.

Informationsdialog

Durch einen Klick auf die Schaltfläche *About* wird nochmals in einem kleinen Fenster das Startbild des *Plugins* zur renalen Funktionsanalyse dargestellt, siehe Abbildung 10.1. Im Gegensatz zu dem beim Aufruf des *Plugins* eingeblendeten Startbild schließt sich dieses Fenster nicht nach drei Sekunden automatisch, sondern wartet auf eine Aktion des Benutzers. Durch einen Klick auf *OK* wird das Fenster wieder geschlossen.

11 Ergebnisse

Der Versuch, ein System zur Untersuchung der funktionellen Nierenperfusion zu entwickeln, kann als durchaus gelungen bezeichnet werden. Besonders erfreulich ist dabei, daß ein Programm entstanden ist, das seine Praxistauglichkeit in Tests im Klinikum Wuppertal unter Beweis gestellt hat und somit das Stadium eines Prototypen bereits hinter sich gelassen hat. Der diagnostische Einsatz des Systems wird auch über den Zeitrahmen dieser Arbeit hinaus andauern und ermöglicht eine nicht unerhebliche Zeitersparnis bei der Diagnosestellung.

Bisher wurden durch Herrn Dr. Stattaus Aufnahmen von 36 Patienten, darunter ein Dialysepatient, mit dem System erfolgreich bearbeitet. Während der Implementierungsphase standen 43 Sequenzen vergangener Untersuchungen zur Verfügung, um die automatische Korrektur der Atmungsbewegung zu testen. Für 14 dieser Serien lagen zusätzlich die manuell erstellten Meßreihen vor und konnten mit den vom System automatisch erstellten Werten verglichen werden, um die Richtigkeit der Ergebnisse zu überprüfen. Für die Kurven des Gesamtorgans zeigte sich dabei in allen Fällen eine sehr gute Übereinstimmung; Unterschiede waren teilweise nur in den Nachkommastellen festzustellen. Da die für Nierenrinde und Nierenmark eingezeichneten ROIs bei der manuellen Auswertung die Nierenfläche nicht abdecken, sind hier die Meßergebnisse nicht direkt vergleichbar. Der Kurvenverlauf und insbesondere der Zeitpunkt des Signalanstiegs und des Maximums stimmten aber auch hier mit den manuell ermittelten Werten überein. Des weiteren konnten drei Sequenzen gefunden werden, deren Atmungsbewegung sich durch die Kepstrum-Filterung nicht korrigieren läßt, weswegen keine weitere Berechnung von SIT-Kurven durchgeführt werden konnte.

Zusammenfassung

Durch die Integration von *Plugins* in das bestehende Bildverarbeitungswerkzeug IMAGEJ ist ein System entstanden, das einen kompletten Arbeitsplatz zur Untersuchung der funktionellen Nierenperfusion anhand kontrastmittel-unterstützter MRT-Turbo-Flash-Sequenzen bereitstellt.

Die im DICOM-Format vorliegenden Sequenzen können importiert und als Bilderstapel dargestellt werden. Die Anwendung der Kepstrum-Filterung ermöglicht eine Korrektur der atmungsbedingten Organverschiebungen und arbeitet dank eingesetzter Interpolationsverfahren sogar im Subpixel-Bereich. Auf den bewegungskorrigierten Sequenzen läßt sich die Signal-Intensitäts-Zeit-Kurve für das Gesamtorgan durch Projektion der ROI(s) leicht berechnen. Das Ausnutzen von a-priori-Wissen über den Kurvenverlauf ermöglicht schließlich eine automatische Klassifizierung der Pixel der Gesamtniere in die Gewebetypen Nierenmark, Nierenrinde und Nierenbecken. Die so berechneten Ergebniskurven können dargestellt und für weitere Auswertungen exportiert werden.

Zusätzlich ermöglicht das System eine Nachjustierung der Klassifizierungsergebnisse, um den anatomischen Varianzen des Bildmaterials gerecht zu werden. Somit können falsche Ergebnisse durch Klassifizierungsfehler direkt visuell durch den Diagnostiker korrigiert werden.

Auch bei problematischen Sequenzen, die unzureichende Kepstrum-Ergebnisse oder nicht justierbare Klassifizierungsfehler aufweisen, können trotzdem die gewünschten Kurven berechnet werden. Einzelne Fehler bei der Kepstrum-Filterung können durch die manuelle Justierung von Halbbildern korrigiert werden. In Ausnahmefällen kann durch das Einzeichnen mehrerer, gewebespezifischer ROIs eine Lösung auf dem konventionellen Bearbeitungsweg erreicht werden.

Die zur Lösung der Problemstellung eingesetzten Methoden werden sich meist nicht in Standard-Lehrwerken zur digitalen Bildverarbeitung finden lassen. Damit zeigt diese Untersuchung, daß insbesondere im Bereich der Verarbeitung medizinischen Bildmaterials eine flexiblere Denkweise zur Problemlösung notwendig ist. Der Einsatz einer angepaßten, zweidimensionalen Kepstrum-Filterung zur Korrektur der Atmungsbewegungen zeigt dies eindrucksvoll. Durch seine Unabhängigkeit von

vorgegebener Objektform, -größe und -lage kompensiert dieses Verfahren die große (anatomische) Varianz des Bildmaterials und liefert gleichzeitig sehr gute Ergebnisse.

Darüber hinaus zeigt das zur automatischen Gewebedifferenzierung entwickelte Verfahren den wichtigen Stellenwert der Wissensakquisition bei fachübergreifenden Arbeiten. Ohne die genauen Aussagen der Mediziner über das Signalverhalten der verschiedenen Gewebearten der Nieren während der Turbo-Flash-Sequenz wäre die Idee zur Untersuchung der SIT-Kurve des Gesamtorgans sicherlich nicht entstanden.

Ausblick

Als wünschenswerte Erweiterung wäre sicherlich eine zuverlässige, automatische Detektion der Organkontur der Niere zu nennen. Dadurch könnte die z.Zt. noch erforderliche Benutzerinteraktion bis hin zu einem automatisch arbeitenden Analysewerkzeug reduziert werden. In diesem Zusammenhang ist der Einsatz intelligenter, wissensbasierter Bildverarbeitungsmethoden zu nennen, da die Segmentierung der Nieren mit Standardverfahren nicht lösbar sein wird [HILT99], [GRAH97], [LIED89].

Zur Steigerung der Geschwindigkeit sollte die bei der Kepstrum-Filterung eingesetzte Fourier-Transformation durch die in Kapitel 7.5 theoretisch behandelte Hartley-Transformation ersetzt werden.

Weiter wäre es sicherlich möglich, die externe Weiterverarbeitung der Kurvendaten (mit einem Tabellenkalkulationsprogramm) ebenfalls in das Programm zu verlagern und das System dadurch mit diagnostischer Intelligenz auszustatten.

Das in Kapitel 7.7.2 diskutierte Problem der unterschiedlichen Anschnitte der Niere aufgrund einer zu großen Abweichung der MRT-Schnittachse von der Achse der Atmungsverschiebung ist sicherlich einfach zu lösen. Eine entsprechende Einstellung des Magnetresonanztomographen vor der Bildakquisition garantiert Sequenzen, die mit der Kepstrum-Filterung gut zu korrigieren sind.

Denkbar wäre sicherlich auch ein in das Programm integriertes Hilfesystem, das Benutzerfragen zur Bedienung beantworten kann. Da im wesentlichen aber immer derselbe Bearbeitungsweg gegangen wird, ist lediglich während einer Einarbeitungsphase Hilfestellung nötig. Hier sollte die Bedienungsanleitung in Kapitel 10.2 ausreichen.

Anhang A: Literaturverzeichnis

[ABRA91] Abramowski S, Müller H: „*Geometrisches Modellieren*“, BI Wissenschaftsverlag, Mannheim 1991

[BAND92] Bandemer H, Gottwald S: „*Einführung in Fuzzy-Methoden*“, 3. Auflage, Akademie Verlag, Berlin, 1992

[BENN97] Bennett HF, Li D: „*MR Imaging of renal function*“, MRI Clinics of North America, 5(1), Seite 107-126, 1997

[BOGE63] Bogert BP, Healy MJR, Tukey JW: „*The Quefrency Alanysis of Time Series for Echoes: Cepstrum, Pseudo-Autovariance, Cross-Cepstrum and Saphe Cracking*“, Proceedings of the Symposium of Time Series Analysis, Seite 209-242, 1963

[BRAC84] Bracewell RN: „*The Fast Hartley Transform*“, Proceedings IEEE 72(8), Seite 1010-1018, 1984

[BRON95] Bronstein IN, Semendjajew KA, Musiol G, Mühling H: „*Taschenbuch der Mathematik*“, Verlag Harri Deutsch, Thun und Frankfurt am Main, 1994

[BUBE93] Bubeck B: „*Die Captopril-Szintigraphie: Ein nuklearmedizinischer Beitrag zur Diagnostik der arteriellen Hypertonie.*“ In: Bubeck B (Hrsg). „*Aktueller Stand der Nierendiagnostik mit ^{99m}Tc MAG3.*“, Hans Huber Verlag, Bern, 1993

[CIDE95] Cideciyan AV: „*Registration of Ocular Fundus Images*“, IEEE Magazine EMB, Band EMB-14, Nr. 1, Seite 52-58, 1995.

[DICOM98] National Electrical Manufacturers Association: „*Digital Imaging and Communications in Medizine* (*DICOM*)“, Teil 1-14, National Electrical Manufactures Association, Rosslyn (Virginia), 1998

[ENCA98] Encarnacao J, Straßer W, Klein R: „*Graphische Datenverarbeitung*“, 4. Auflage, Oldenburg Wissenschaftsverlag, München, 1998

[FELI97] Felix R, Heshiki A, Hosten N, Hricak H: „*Magnevist – Eine Monographie*“, 2. Auflage, Blackwell Wissenschaftsverlang Berlin 1997

[FELL92] Fellner WD: „*Computergrafik*“, 2. Auflage, BI Wissenschaftsverlag, Mannheim 1992

[FOLE97] Foley JD, van Dam A, Feiner SK, Hughes FJ: „*Computer Graphics - Principles and Practice*“. Addison-Wesley Verlag, 1997.

[GEIS98] Geis T, Hartwig R: „*Auf die Finger geschaut. Neue ISO-Norm für benutzergerechte interaktive Systeme zur DIN EN VO 13407*“, c't Magazin für Computertechnik 14/98, Seite 168-171, Heise Verlag, Hannover, 1998

[GERI91] Gerig G, Kikinis R, Kuoni W, von Schulthess GK, Kübler O: „*Semiautomated ROI Analysis in Dynamic MR Studies - A Tool for Automatic Correction of Organ Displacements*“, Journal of Computer Assisted Tomography 15(5), Seite 725-732, Raven Press, New York, 1991

[GERL88] Gerlot P, Bizais Y: „*Image Registration: A Review and a Strategy for Medical Applications.*“, De Draaf DN, Viegever MA (editors), Proc. of the 10th Intern. Conf. on Information Processing in Medical Imaging, Plenum Press, New York, Seite 81-89, 1988.

[GOER95] Goerke C: „*Geometrischer Angleich digitaler Freihand-Radiographien durch Kepstrum- und Spektrum-Techniken*“ Diplomarbeit, Institut für Biometrie und

medizinische Informatik, RWTH Aachen, 1995

[GONZ92] Gonzales RC, Woods RE: „*Digital Image Processing*“, Addison-Wesley Verlag, 1992

[GRAH97] Graham D, Barrett A: „*Knowledge-Based Image Processing Systems*“, Applied Computing, Band 13, Springer-Verlag, London, 1997.

[HABE95] Haberäcker P: „*Praxis der digitalen Bildverarbeitung und Mustererkennung*“, Hanser-Verlag, München, Wien, 1995

[HACK97] Hackländer T: „*Kernspintomographische Messung des regionalen cerebralen Blutvolumens nach der Relaxationsmethode*“, Habilitationsschrift, Medizinische Fakultät, Heinrich-Heine-Universität Düsseldorf, 1996.

[HACK99] Hackländer T: Java-Portierung einer Dicom-Library, Dicom-Import-*Plugin* für IMAGEJ, Implementierung, 1999 URL:

`http://www.klinikum-wuppertal.de/kliniken/radiologie/pages/forschung.html`

[HARR78] Harris FJ, „*On the Use of Windows for Harmonic Analysis with the discrete Fourier-Transform*“ Proceedings IEEE 66(1), Seite 51-83, 1978

[HART42] Hartley RVL: „*A more symmetrical Fourier Analysis applied to Transmission Problems*“, Proceedings IRE 30(3), Seite 144-150, 1942

[HASH97] Hashemi RH, Bradley WG: „*MRI - The Basics*“, Williams & Wilkins Verlag, Baltimore, 1997

[HER94] Her I, Yuan CT: „*Resampling on a pseudoheaxgonel Grid*“ CVGIP: Graphical Models and Image Processing 56(4), Seite 336-347, 1994

[HFR99] Hiltner J, Fathi M, Reusch B (Herausgeber): Projektgruppe 331, „*Dawn Endbericht – Entwicklung eines wissensgesteuerten Bildverarbeitungssystems zur Erkennung von Verkalkungen in Koronararterien*“, Lehrstuhl Informatik I, Universität Dortmund, 1999

[HILT99] Hiltner J: „*Einsatz von Methoden der Computational Intelligence in der digitalen Bildverarbeitung*“ In: B. Reusch (Herausgeber): Interdisziplinäre Methoden der Informatik, Forschungsbericht, Universität Dortmund, Januar 1999.

[HORN84] Horner JL, Gianino PD: „*Phase-only Matched Filter*“, Applied optics 23(6), Seite 812-816, 1984

[HORN98] Horn C, Kerner IO: „*Lehr und Übungsbuch Informatik – Band 4: Technische Informatik und Systemgestaltung*“, Kapitel „*Entwicklung grafischer Benutzer-oberflächen*“, Seite 239-274, Fachbuchverlag Leipzig, Carl Hanser Verlag, München, 1998

[JÄHN97] Jähne B: „*Digitale Bildverarbeitung*“, 4. Auflage, Springer Verlag, Berlin, 1997.

[KAMM92] Kammeyer KD, Kroschel G: „*Digitale Signalverarbeitung*“, B.G. Teubner Verlag, Stuttgart, 1992.

[KASS87] Kass M, Witkin A, Terzopoulos D: „*Snakes: Active Contour Models*“, International Journal of Computer Vision, Band 3, Seite 259-268, 1987.

[KEYS81] Keys RG: „*Cubic Convolution Interpretation for Digital Image Processing*“, IEEE ASSP-29(6) Seite 1153-1160, 1981

[KLET95] Klette R, Zamperoni P: „*Handbuch der Operatoren für die Bildverarbeitung*“, 2. Auflage, Vieweg Verlag, 1995

[KNOO97] Knoop MV, Dörsam J, Oesingmann N, Piesche S, Hawighorst H, Wiesel M, Schad

LR, van Kaick G: „*Funktionelle MR-Urographie bei Patienten mit Nierentransplantaten*“, Radiologie 37, Seite 233-238. Springer Verlag, Berlin, 1997

[KOPP97] Kopp H: „*Bildverarbeitung interaktiv*“. B.G. Teubner Verlag, Stuttgart, 1997.

[KRES88] Krestel E: „*Bildgebende Systeme für die medizinische Diagnostik*“, 2. Auflage, Siemens AG, Hrsg. Heinz Marneburg, 1988

[LAUB90] Laubenberger T: „*Technik der medizinischen Radiologie*“, 5. Überarbeitete Auflage, Dt. Ärzte-Verlag GmbH, Köln, 1990

[LEE89] Lee DJ, Mitra S, Krile TF: „*Analysis of sequential complex images using feature extraction and two-dimensional cepstrum techniques*“ JOSA A-6(6), Seite 863-870, 1989

[LEHM97] Lehmann T, Oberschelp W, Pelikan E, Repges R: „*Bildverarbeitung für die Medizin. Grundlagen, Modelle, Methoden, Anwendungen*“ 1. Auflage, Springer Verlag, Berlin, 1997.

[LEHM98] Lehmann T: „*Geometrische Ausrichtung medizinischer Bilder am Beispiel intraoraler Radiographien*“, Dissertationsschrift, Institut für medizinische Informatik, RWTH Aachen, 1998

[LIED89] Liedtke CE, Ender M: „Wissensbasierte Bildverarbeitung“, Reihe Nachrichtentechnik, Band 19, Springer Verlag, Berlin/Heidelberg, 1989

[LIND89] Linder M: „*Biologie – Lehrbuch für die Oberstufe*“, J.B. Metzler Verlag, Stuttgart, 1989

[LISS90] Lissner J, Seiderer M: „*Klinische Kernspintomographie*“, 2. Auflage, Enke Verlag, Stuttgart, 1990

[LYON99] Lyon DA: „*Image processing in Java*“, Prentice Hall Verlag, 1999

[MART00a] Martin J, Hiltner J, Fathi M, Reusch B, Stattaus J, Hackländer T: „*A system for renal movement elimination and renal diagnosis supported by vague knowledge*“ SPIE Symp. on Medical Imaging 3979(141) S18, San Diego, CA, USA, 2000

[MART00b] Martin J, Hiltner J, Reusch B, Stattaus J, Hackländer T: „*Ein System zur funktionellen Perfusionsuntersuchung der Niere bei gleichzeitiger Unterdrückung der Atembewegung des Organs*“ In: Bildverarbeitung für die Medizin 2000, Informatik aktuell, Springer Verlag, München, März 2000 (akzeptiert)

[MILL99] Miller S, Hahn U, Schick F, Nägele T, Duda SH, Claussen CD: „*Diagnostik von Nierenarterienstenosen bei 1,0T mittels 3D-Phasenkontrast-MR-Angiographie und dynamischer Kontrastmittelflutung*“ Fortsch. Röntgenstr. 170, Seite 163-167, Thieme Verlag, 1999

[MORN95] Morneburg H: „*Bildgebende Systeme für die medizinische Diagnostik*“, 3. Auflage, Siemens AG, 1995

[MÜLL95] Müller H: „*Digitale Bildverarbeitung*“, Skriptum zur Vorlesung, Universtät Dortmund, 1995

[NICO97] Nico N, Albrecht R: „*Wissenschaftliche Arbeiten schreiben mit Winword 97*“, Addison Wesley Verlag, Bonn, 1997

[OPPE75] Oppenheim AV, Schäfer RW: „*Digital Signal Processi*ng“ Prentice-Hall, Englewood Cliffs, NJ, 1975

[PARK83] Park SK, Schowengerdt RA: „*Image Reconstruction by Parametric Cubic Convolution*“, CVGIP 23, Seite 258-272, 1983

[PENG91] Penglin H, Yuanlong C, Dequn L: „*On Cepstrum Approach to Sequential Image Matching*“, IEEE Conference Proceedings, China 1991 International Conference on Circuits and Systems, Shenzhen/China, 2: Seite 797-800, 1991

[PHIL95] Philips: „*Basic Principles of MR Imaging*“, Medical Systems, Niederlande, 1995

[PSCH98] Pschyrembel: „*Klinisches Wörterbuch*“, 258. Auflage, de Gruyter Verlag, 1998

[PYKE82] Pykett IL: „*Kernspintomographie – Röntgenbilder ohne Röntgenstrahlen*“, Spektrum der Wissenschaft, Seite 40-54, Juli 1982

[RASB98] Rasband W: IMAGEJ: Ein JAVA-basiertes Bildverarbeitungsprogramm, URL: `http://rsb.info.nih.gov/ij/index.html`, Implementierung, 1998-2000

[REIS97] Reiser M, Semmler W: „*Magnetresonanztomographie*“, 2. Auflage, Springer Verlag, 1997

[REVE97] Revet B: „*DICOM Cook Book for Implementations in Modalities*“, Philips Medical Systems, Niederlande, 1997

[REVO97] Revol C, Jourlin M: „*A new minimum variance region growing algorithm for image Segmentation*“, Pattern Recognition Letters 18, Seite 249-258, 1997

[RFH98] Reusch B, Fathi M, Hiltner J (Herausgeber): Projektgruppe 305, „*Anomalia-Endbericht – Ein Wissensgesteuertes Bildverarbeitungssystem zur Erkennung pathologischer Strukturen in medizinischen Bilddaten*“, Lehrstuhl Informatik I, Universität Dortmund, 1998

[ROCH96] Roche „*Lexikon der Medizin*“, Urban & Schwarzenberg Verlag, München, 1996

[SCHA99] Schalla C, Trümper A: „*Entwicklung eines virtuellen klinischen Kernspintomographen*“, Diplomarbeit, Universität Dortmund, 1999

[SCHU91] von Schulthess GK, Kuoni W, Gerig G, Wüthrich R, Duewell S, Krestin,G: „*Semiautomated ROI Analysis in Dynamic MR Studies - Application to Renal Function Examination*“, Journal of Computer Assisted Tomography 15(5), Seite 733-741, Raven Press, New York, 1991

[SIEM92] Siemens AG: „*Magnet, Spins und Resonanzen – Eine Einführung in die Grundlagen der Kernspintomographie*“, Erlagen, 1992

[STAT99] Stattaus J, Mertens H, Hackländer T, Altenburg A, Cramer BM: „*MRT in der Diagnostik der Nierenarterienstenosen: Kombination aus KM-verstärkter MR-Angiographie und dynamischer MRT*.“ Fortsch. Röntgenstr. 170, Seite 115, Thieme Verlag, 1999

[STEC89] Steckner MC, Drost DJ: „*Fast Cepstrum Analysis using the Hartley Transform*“, IEEE ASSP-37(8), Seite 1300-1302, 1989

[STUE99] Stüttgen T: „*Bildsegmentierung mit aktiven Konturmodellen unter Berücksichtigung unscharfen Wissens am Beispiel medizinischer Bilddaten*“, Diplomarbeit, Universität Dortmund, 1999.

[SUN99] Sun Microsystems: „*Java Look and Feel Design Guidelines*“, Sun Microsystems, Palo Alto (Kalifornien), 1999

[THEW91] Thews G, Mutschler E, Vaupel P: „*Anatomie und Pathophysiologie des Menschen*“, WVG, Stuttgart, 1991

[VISI95] The Visible Human Project, U.S. National Library of Medicine, Rockville, USA, 1995

[WAHL89] Wahl FM: „*Digitale Bildsignalverarbeitung*“, 2. Auflage, berichtigter Nachdruck,

Springer Verlag, 1989

[WASS97] Wasser MN, Westenberg J, van der Hulst VPM, van Baalen J, van Bockel JH, van Erkel AR, Pattynama PMT: „*Hemodynamic significance of renal artey stenosis: digital subtraction angiographie versus systolically gated three-dimensional phase-contrast MR angiographie*“ ,Radiology 1997(202), Seite 333-338, 1997

[WATS86] Watson AB, Poirson A: „*Seperable Two-Dimensional Discrete Hartley Transform*“, JOSA A-3(12), Seite 2001-2004, 1986

[WILH99] Wilhelms G, Kopp M.: „*Java Professionell*“, MITP-Verlag, Bonn, 1999

[WOLB90] Wolberg G: „*Digital Image Warping*“, IEEE Computer Society Press, 1990

Anhang B: Abbildungs- und Tabellenverzeichnis

Abbildung 1.1: Beispiel einer MRT-Nieren-Serie. 5
Abbildung 2.1: Bauchraum des Menschen. 7
Abbildung 2.2: Rechte Niere in Vorderansicht (a), schematische Ansicht der Lage (b). 8
Abbildung 2.3: Anschnitt der rechten Niere (a) und Aufbau eines *Nephrons* (b). 9
Abbildung 2.4: Nierenkarzinom (a), Zystenniere (b), Schrumpfniere (c). 11
Abbildung 2.5: Wirkungsweise von *Renin* an den *Glomeruli*. 12
Abbildung 2.6: Die komplexen Zusammenhänge der menschlichen Blutdruckregulierung. 13
Abbildung 2.7: Hypertonischer Bereich und Grenzbereich des Blutdrucks. 13
Abbildung 2.8: Digitale Röntgen-Subtraktionsangiographie. 15
Abbildung 3.1: Röntgenaufnahme, Ultraschall, CT-Aufnahme eines Kopfes, MRT-Aufnahme eines Kopfes. 17
Abbildung 3.2: Kernspin und Präzession (a), Ausrichtung der Kernspins im Magnetfeld (b). 19
Abbildung 3.3: Blockschaltbild eines MRT (a), MRT-Gerät (b). 19
Abbildung 3.4: Auswirkung eines 90°- und eines 180°-Impulses auf den Kernspin. 21
Abbildung 3.5: Konturbetonung an der Grenze zwischen Nieren und umgebendem Gewebe. 22
Abbildung 3.6: Fenstern. 23
Abbildung 4.1: IMAGEJ (a) und eine Nierensequenz als Bilderserie (b). 25
Abbildung 4.2: Verwendete *Plugins* von IMAGEJ und deren Aufruf. 26
Abbildung 4.3: Anordnung von Referenzbild und Folgebild im Keptrum-Fenster. 27
Abbildung 4.4: Peaks in der Kepstral-Ebene. 28
Abbildung 4.5: Arbeitsweise des Algorithmus zur Translationskorrektur. 29
Abbildung 4.6: Meßkurve einer Beispielsequenz für die gesamte Niere (Bilder 1-30). 29
Abbildung 4.7: Automatisch berechnete Meßkurven für eine Beispielsequenz (Bilder 1-40). 31
Abbildung 4.8: Manuelles Einzeichnen multipler ROIs. 32
Abbildung 4.9: Übersicht über die Komponenten des Systems sowie den Bearbeitungsweg. 33
Abbildung 5.1: Beispielbild zur Veranschaulichung verschiedener Bildverarbeitungsoperatoren. 34
Abbildung 5.2: Porträtaufnahme (zu dunkel) mit zugehörigem Histogramm. 36
Abbildung 5.3: Siehe Abbildung 5.2a, mit modifizierter Grauwertverteilung. 37
Abbildung 5.4: Beispiel zur Bildung des Differenzbildes (c) durch Subtraktion des Bildes (b) von (a). 38
Abbildung 5.5: Prinzip lokaler Operatoren im Ortsraum. 39
Abbildung 5.6: gestörtes Bild (a), mit lokalem Tiefpaßfilter geglättet (b), mit Median-Filter geglättet (c). 40
Abbildung 5.7: Prinzip lokaler Gradientenoperatoren. 40
Abbildung 5.8: Beispielbild (a), Anwendung des Sobel-Operators (b). 41
Abbildung 5.9: Beispiele zum Verständnis der Fourier-Transformation. 44
Abbildung 5.10: Darstellung des Betragsspektrums der Fourier-Transformatierten. 44
Abbildung 5.11: Prinzip der Filterung im Frequenzbereich. 46
Abbildung 5.12: Berechnung der kubischen Interpolation. 50
Abbildung 6.1: Beispiel zur Korrelation im Frequenzraum. 56
Abbildung 6.2: Betrag (a) und Phase (b) des Korrelationssignales $F_1^* \cdot F_2$. 57
Abbildung 6.3: Ein 64×64 Pixel großer Ausschnitt aus der Rücktransformierten der korrelierten Bilder. 57
Abbildung 6.4: Durchführung der Kepstrum-Filterung c_{KEP} mit D=0. 60
Abbildung 6.5: Ergebnis der einfachen Kepstrum-Filterung c_{KEP} (a) und der erweiterten Kepstrum-Filterung c_{DKEP} (b). 61
Abbildung 6.6: Anordnung der Bilder nebeneinander im Kepstrum-Fenster. 62
Abbildung 6.7: Wiederholung der Maxima bei der diskreten Kepstrum-Filterung. 62
Abbildung 7.1: Prinzip der Kepstrum-Filterung. 66
Abbildung 7.2: Beispiel zum Histogrammangleich. 67
Abbildung 7.3: Beispiel zu inhomogenen Gewebeverschiebungen. 69
Abbildung 7.4: Nichtlineare Gewebeverschiebungen, Differenzbilder von 6 Beispielserien (a)-(f). 69
Abbildung 7.5: Peak in der Kepstral-Ebene nach Vorverarbeitung. 70
Abbildung 7.6: Die zur mittleren Zeile des Fensters aus (7.8) gehörenden Differenzbilder. 71
Abbildung 7.7: Unterschiedliche Anschnitte der Nieren in aufeinanderfolgenden Bildern. 72
Abbildung 7.8: „Wandern“ der Nieren durch sich fortsetzende Fehler bei der Kepstrum-Filterung. 73
Abbildung 7.9: Kleineres Template aus dem justierten Folgebild und Suchbereich des Referenzbildes. 73
Abbildung 7.10: (Ausgesuchte) Beispiele für problematische Nieren-Sequenzen. 75
Abbildung 7.11: Ergebnisse der automatischen Detektion der Nierenkontur. 75
Abbildung 8.1: Markierte Gesamt-ROIs (a), Signal-Intensitäts-Zeit-Kurven der gesamten Nieren (b). 76
Abbildung 8.2: Darstellung der vier Hilfskriterien zur Klassifizierung (als anschauliche Grauwertbilder). 79
Abbildung 8.3: Ergebnis der Gewebeklassifizierung für die linke und rechte Niere einer Beispielserie. 80
Abbildung 9.1: Ein StackWindow-Objekt mit Schieberegler. Der ImageCanvas stellt das Bild dar. 88

Abbildung 9.2: Zuordnung der Pakete zu den Programmkomponenten. 89
Abbildung 9.3: Klassendiagramm der zum Import benötigten Klassen. 90
Abbildung 9.4: Übersicht der für die Realisierung des *Plugins* zur renalen Funktionsanalyse benötigten Klassen. 95
Abbildung 9.5: Die zur Kepstrum-Filterung benötigten Klassen. 99
Abbildung 9.6: Klassendiagramm für die SIT-Kurven-Berechnung. 101
Abbildung 9.7: Darstellung der zur manuellen (Nach-)Bearbeitung benötigten Klassen. 104
Abbildung 9.8: Ein *JLabelResult*. Links oben werden die Werte für die Kurve an der Mausposition angezeigt. 106
Abbildung 9.9: Die zur Ergebnisdarstellung benötigten Klassen. 107
Abbildung 9.10: Markierte Scanline (a), Intensitätsprofil der Scanline (b), ausgehend vom Mittelpunkt der Niere. 108
Abbildung 10.1: Titelbild des IMAGEJ-*Plugins* zur renalen Funktionsanalyse. 110
Abbildung 10.2: IMAGEJ-Hauptfenster. 112
Abbildung 10.3: Dialog zum Import der Nierensequenzen. 113
Abbildung 10.4: Ein Dateiauswahldialog (engl. File-Open-Dialog). 113
Abbildung 10.5: Dialog zum Import der Nierensequenzen. 114
Abbildung 10.6: Fortschrittsanzeige in der Statuszeile von IMAGEJ. 114
Abbildung 10.7: Darstellung der importierten Nierensequenz als Bildstapel. 115
Abbildung 10.8: Hauptfenster des *Plugins* zur renalen Funktionsanalyse. 116
Abbildung 10.9: Fenster mit Nierensequenz. Erweiterte Darstellung. 116
Abbildung 10.10: Fehlermeldungen bei fehlenden Properties oder fehlenden Bildsequenzen. 117
Abbildung 10.11: IMAGEJ-Schaltfläche für rechteckige ROIs. 118
Abbildung 10.12: Markieren rechteckiger ROIs. 118
Abbildung 10.13: Markieren einer Polygon-ROI. 119
Abbildung 10.14: Anzeige der Fortschritts der Kepstrum-Filterung im IMAGEJ-Fenster. 120
Abbildung 10.15: Meldungen nach Beendigung der Kepstrum-Filterung. 120
Abbildung 10.16: Hilfsdialog zur manuellen Justierung der Verschiebung. 122
Abbildung 10.17: Ergebnisfenster der SIT-Kurven-Berechnung. 123
Abbildung 10.18: Berechnete Werte an der Mausposition (a), Darstellung der Standardabweichung (b). 123
Abbildung 10.19: Dialog zum Export der berechneten Kurvenwerte als Textdatei. 124
Abbildung 10.20: Dialog zur Justierung der Klassifizierungsergebnisse. 125
Abbildung 10.21: Abfrage beim Festsetzen von gewebespezifischen ROIs. 126
Abbildung 10.22: Einzeichnen mehrerer gewebespezifischer ROIs (gelb, rot), Gesamt-ROI (grün). 126
Abbildung 10.23: Die Karteikarte *Cepstrum Settings*. 128
Abbildung 10.24: Die Karteikarte *SIT Settings*. 129
Abbildung 10.25: Die Karteikarte *Other Settings*. 130

Tabelle 3.1: Darstellungsmöglichkeiten der 4 wichtigsten bildgebenden Verfahren in der Medizin. 18
Tabelle 3.2: Signalintensität verschiedener Strukturen in T_1- und T_2-gewichteten Bildern. 20
Tabelle 3.3: Gradientenverfahren. Akronyme einiger Gerätehersteller. 22
Tabelle 6.1 Die Kunstworte zur Unterscheidung von Spektral- und Kepstral-Ebene. 59
Tabelle 7.1: Mittlerer Grauwert (g_{av}) und Standardabweichung (g_{dev}) der Beispielbilder aus Abbildung 7.4. 70
Tabelle 8.1: Erklärung der wichtigen Bezeichner für dises Kapitel. 77
Tabelle 9.1: Import der vom System exportierten Berechnungsdaten mit einem Tabellenkalkulationsprogramm. 105
Tabelle 10.1: Beschreibung der auf der Karteikarte *Cepstrum Settings* wählbaren Voreinstellungen. 129
Tabelle 10.2: Beschreibung der auf der Karteikarte *SIT Settings* wählbaren Voreinstellungen. 130
Tabelle 10.3: Beschreibung der auf der Karteikarte *Other Settings* wählbaren Voreinstellungen. 130

Anhang C: Index

1

180° HF-Impuls 19
180°-Inversions-Präparations-Impuls 20

9

90° HF-Impuls 19

A

Abdomen 8
Abdomenbereich 23
Abfragedialog 120
Ableitung 74
Abtasttheorem 43
addRoiToArray 97
Adiuretin 11

Ä

Ähnlichkeitssatz 40

A

Akquisitionsintervall 23
Akquisitionszeit 20
Aldosteron 11
Algorithmus 26
ALT-Taste 88, 97, 111
Amplitude 52
Amplitudenspektrum 38
AnalysisDialog 86
angepaßter Filter 51
Angiographie 13
Angiotensin 11
anguliert 23
Anleitung.pdf 104
Antennensystem 18
Anzeigefläche 111
append 93
applyShift 92
applyWindow 92
a-priori-Wissen 27
Arbeitsablauf 26
Arbeitsplatz 96, 119
arg 81
Array 86, 88
Arretierung 114
artifizielle Konturbetonung 70
Atmungsbewegungen 67
Atmungsverschiebung 72, 77
Atmungsverschiebungen 24
Atomkerne 17
Attribute 82
Auflösung 64
Aufnahmegeometrie 13, 64, 67
Aufnahmeinformationen 110
Aufnahmeparameter 86
Ausführungsgeschwindigkeit 78
Auslesegradient 18
Autokorrelationsfunktion 49
automatische Segmentierung 70
AWT 83, 89
AwtToolTip 89
awtToolTip.showTooltip 89
AwtToolTipHelper 89

B

Basisfunktionen 43
Basismatrizen 44
Bauchraum 8
Bearbeitungsweg 29
Bedienungsanleitung 104
Befundunterstützung 48
Beispielsequenz 27
Benutzeroberfläche 104
Benutzeroberflächen 78
Benutzerschnittstelle 84
Berechnungsklasse 94, 101
Bereichswachstumsverfahren 70
Betragsgradientenbild 36
Betragsspektrum 38
Betriebssystem 78
Bewegungskorrektur 28
bewegungskorrigierte Bilddaten 26
Bewertungskriterien 26
Bicubic_Scaler 103
BicubicInterpolation 91, 92
bikubische Interpolation 77
Bilddimensionen 103
Bilderstapel 23, 73, 81, 109
Bildfunktion 31, 73
Bildgewinnung 35
Bildgewinnungzyklus 20
Bildgröße 23
Bildnachverarbeitung 90
Bildproportionen 103
Bildpunkt 31, 73
Bildpunktmenge 73
Bildregion 25
Bildregistrierung 24
Bildskalierung 77, 81
Bildverarbeitung 22, 81
Bildverarbeitungsprogramm 23
Bildverarbeitungssysteme 48
Bildverbesserung 33
bilineare Interpolation 44
Bluthochdruck 12
Blutvolumenregulierung 11
BOLTZMANN-Verteilung 18
Browser 78
Bytecode 78

C

calcAngle 103
calcDistance 103
calcPixelCurve 95
calculateTranslation 92
Canvas 88
Cepstrum 91
Cepstrum.log 93
CepstrumAnalysis 91
cepstrumFinished 87
ChangeFileAttributes 105
charakteristischer Signalanstieg 72

checkCepstrumResult 92
classifyPixels 95, 101
ComplexImage 90
Computertomographie 16
Cortex 8, 76, 97
Creatinin 15
CT 16

D

Dämpfung 20
Darstellungen 99
Dateiauswahldialog 107, 117
Dateiname 99
Datenexport 29
Datenimport 107
dcm-Paket 84
Debuggen 78
deckungsgleich 24
Deckungsgleichheit 69, 114
Deformation 68
Deformations-Raster 69
derivation 95
DHT 63
Diagnosestellung 76
Diagnoseunterstützung 105
Diagnostiker 28
DICOM 23
DICOM-Import 23
Dicom-Importfilter 82
Dicom-Standard 81
Differenzbild 33, 50
Differenz-Filterung 122
Differenzierung 74
digitale Bildverarbeitung 30
Digitalisierung 37
Dilatation 13
Dirac-Impulse 56
Dirac-Stoß 57
disjunkte Mengen 74
Disjunktion 75
diskreter Bereich 31
Diskretisierung 37
Distanzbild 75
Diurese 7
Diuretika 11
doCepstrum 92
Doppelsumme 42
doSegmentation 102
Drehwinkel 25
dreidimensionale Bilddaten 31

E

Echolaufzeit 54
Echtzeitfähigkeit 17
Echtzeit-Klassifizierungs-Algorithmus 101
Eigendrehimpuls 17
Eigenrotation 18
eindimensionaler Wertebereich 31
eine RST-Invariante 61
Einfluten 72
einseitige Verzögerung 27
Elektrolythaushalt 7
elektromagnetische Wellen 37
EM-Signal 21
ENTF-Taste 88, 111
Ergebniskontrolle 26
Exceptions 78
ExImageCanvas 86, 88
ExImageStack 86
ExImageWindow 86
Exit-Button 87
Exponentialfunktion 20
Export 29, 99, 117
extrazellulares Flüssigkeitsvolumen 11

F

Faltung 51
Faltungskern 34
Faltungsprodukt 41
Faltungssatz 41
Faltungstheorem 51
Farbkodierung 116
Farbschemata 87
Farbverteilung 31
Fast-Fourier-Transformation 61
Faszie 8
f_{diff} 33
Fehlermeldungen 110
Fensterbreite 21
Fensterfunktion 56
Fensterkreuz 59
Fensterung 21, 61
Fett 19
FFT 61, 91
fibröse Plaques 13
field of view 23
fillMissingValues 95
Filter 86
filterSobel 92
Filtrat 14
Filtrationsdruck 9
findLocalMax 92
Fließkomma 81
floatingAverage 99
Flußdiagramm 26
Flußdichte 18
Folgebild 24, 64
Foriertransformationspaar 38
Fourier-Analyse 24
Fourier-Reihe 42
Fourier-Transformation 25, 37, 78
Fourier-Transformierte 38
FOV 23
Freihand-ROI 113
Frequenzbereich 31, 54
Frontalaufnahme 113
Fructosepolysaccharid 15
Funktion 31
Funktionalzusammenhang 34
funktionelle Nierenperfusion 23
funktionelle Perfusionsuntersuchung 72
Fuzzy-Sets 70

G

g_{av} 32
Gd-DTPA 23
g_{dev} 32
Gesamtorgan 26
Gesamtrechenzeit 79
Geschwindigkeitssteigerung 63
getMaxArrayVal 95
getPixel 92
getPixels 82
getProperties 86

getRoi 88
getRotatedSobel 102
Gewebearten 18
Gewebedifferenzierung 119
Gewebegruppen 27
Gewebeklassifizierung 28
Gewebetyp 95
GFR 14
Glätten 99
Glättung 27
gleichverteiltes Histogramm 31
glomeruläre Filtrationsrate 14
Glomeruli 9
Glomerulonephritiden 10
Glomerulussklerose 10
Gradienten-Echo-Sequenz 19
Grafikdateiformate 81
Grafiktablett 113
Grandienten-Echo-Techniken 19
Grauwert 73
Grauwertanstieg 74
Grauwertbereich 21
Grauwertmenge 31
Grauwertschwankungen 52
Grauwertverteilung 31
Grenzwerte 27
Grundfarben 31
GUI 83, 99
GUI-Komponenten 87

H

Halbbild 28
hämatogene Metastasen 10
Harnblase 8
Harnkonzentration 9
Harnleiter 8
Harnzeitvolumen 15
Hartley-Transformation 63
Hashtable 82
Hauptdialog 86
Hauptmaximum 57
Haupt-ROI 97
Hauptspeicher 105
header 29, 79
Helligkeit 21, 31
Herz-Kreislauf-System 9
Herzzeitvolumen 11
HF-Frequenz 18
HF-Puls 18
Hilfsfunktionen 100
Hilfsklassen 84
Hilus 8
histogramEquilisation 92
Histogramm 31, 33
Histogrammangleich 62, 96
Histogrammfunktion 31
Hochskalieren 64
Hypertonie 11, 12

I

IBM JDK 1.1.8 79
ij.jar 106
ij.pro 81
ImageCanvas 83
IMAGEJ 23, 77, 81
IMAGEJ-Hauptfenster 108
ImageLabel 94
ImagePlus 29, 81, 82
ImageProcessor 102
ImageStack 81
ImageWindow 82, 85
Imaginärteil 90
Implementierung 70, 81
Import 81, 84
Importfunktion 81
Induktionssignal 20
Informationsfenster 114
inhomogene Verschiebung 64
inhomogenen Verschiebungen 68
Installation 105
Instanzvariablen 79
int86
Intensität 27, 73
Intensitätsbereich 25
Intensitätsdifferenzen 36
Intensitätsprofil 102
Intentesitätsanstieg 62
Internet 78
interpolateNullValues 95
Interpolation 44, 77
Interpolationsfehler 47
Interpolationsfunktion 45
Interpolationsverfahren 64
Interstitiellen Nephritiden 10
Inulin 15
Inulin-Clearance 15

J

JAVA 23, 77
Java Developer Kit 78
java.awt.Image 91
java.awt.Point 92
java.awt.Polygon 103
java.lang.Thread 92
JAVA-Applet 78
JavaDoc 79
JAVA-Entwicklungsumgebung 78
JAVA-Grafikpaket 83
JavaImageGraphicsLibrary 90
Java-Pakete 83
JBuilder 78
JDK 78
jigl 90
JIT-Compiler 78
JLabelResults 94
JMConst 89
JMUtil 89
JTabbedPanes 87

K

Kaiser-Bessel-Fenster 61
Kalksalzen 13
Kantenextraktion 61
Karteikarte 121
Kepstral-Bereich 25
Kepstral-Ebene 25, 54, 57, 58
Kepstrum 54
Kepstrum-Fenster 25
Kepstrum-Filterung 24, 66
Kepstrum-Technik 24, 48, 54
Kernspin-Resonanz-Signal 18
Kernspintomographie 16, 17
KeyListener 88, 97
kidney 86

Klassen....79
Klassendefinition....79
Klassendiagramm....85
Klassennamen....79
Klassifizierung....77, 101
Klassifizierungsbedingungen....74
Klassifizierungsbilder....101, 116
Klassifizierungskriterien....72
Knochen....17
Knochenmark....19
Knorpel....19
Kommentare....79
Kompaßgradient....37
komplexwertig....38
Komponenten....29
konjugiert Komplexe....51
Konstante....32
Konstruktor....79
kontinuierliche Bilder....31
kontinuierlicher Raum....37
Kontrast....21, 31
Kontrastmittel....13, 23, 27, 73
Kontrastmittelaufnahme....96
Kontrastmittelpassage....19
Kontrastmittelverhalten....26
Kontrastunterschiede....59
Kontrastverstärkung....74
Kontrastwahrnehmung....87
Konturänderungen....70
Konturbetonung....20, 73
Kontur-Polygon....103
Konturpunkt....102, 112
Koordinaten....31
Koordinatensystem....19, 99
koronar....23
Korrelation....51
Korrelationskoeffizienten....49
Kortikalis....19
k-Raum....43
Kreiselbewegung....18
Kreuz-Korrelation....48
Kreuz-Korrelationsfunktion....48
Kreuz-Kovarianzkoeffizient....50
kubische Interpolation....44
kubischen Interpolation....45

L

Lageinformationen....74
langfristige Blutdruckregulation....11
Larmorfrequenz....18
Laufzeitumgebung....106
Leakage-Effekt....61
Leber....8
Lehrbuchcharakter....30
Leistungsspektrum....38
lineare Abbildung....32
lineare Glättungsoperatoren....35
lineare Skalierung....32
Linearfaktoren....43
Linearkombination....44
LINUX....105
Liquor....19
loadPresetsFile....87
lock....88
Log-File....123
lokale Gradientenoperatoren....36
lokale Operatoren....34
Low-Level-Routinen....83
Lupen-Schaltfläche....112
Lymphgefäße....8
Lymphknoten....19

M

MACOS....105
Magnetisierungsvektor....19
Magnetom Vision....23
Magnetresonanztomographie....13, 15, 17
Magnitudenbild....91
ManualMover....86, 88, 97
manualSelector....95
manuelle Auswertung....28
Markschicht....74
Maschinencode....78
Maske....69
matched filter....51
Matrix....34
Mausposition....116
Mauszeiger....112
maximale Anreicherung....27
Median-Filter....35, 102
Median-Filterung....61
Medulla....8, 97
Menüstruktur....81
Methoden....80
Mindestabstand....69
Minimumsuche....66
Mittelpunkt....25
Mittelwertbildung....74
mittlere Grauwert....32
mittlere Intensität....27
mittlerer Grauwert....73
Module....78
Mouse-Handling....83
MouseListner....97

N

Nachjustierung....66, 92, 115
Namensgebung....79
Nephritiden....10
Nephron....8
Netzwerk....23
Nierenangiographie....13
Nierenarterie....8
Nierenarterienstenose....11, 27
Nierenarterienstenosen....72, 76
Nierenbecken....8, 74
Niereninfarkt....10
Nierenkarzinome....10
Nierenkelchen....8
Nierenkontur....88, 102
Nierenkonturen....62
Nierenmark....27, 72, 74, 95
Nierenpapillen....8
Nierenperfusion....14, 24
Nierenpforte....8
Nierenrinde....8, 27, 72, 74, 95
Nierenschädigung....27
Nierenserie....29
Nierenvene....8
Nierenzysten....10, 70
Nukleone....17

O

Oberflächengestaltung....87

Object....82
Objektform....25
Objektkontur....25

Ö

Ödem....19

O

Operatoren....23, 81
Operatorfenster....34
Organfunktionen....17
Organgrenze....70
Organschäden....25
Organverschiebungen....122
Ortsbereich....31
Ortsfrequenzspektrum....38
Ortsfrequenzvariablen....38

P

paint....88
PanelDialog....84
Parenchymschrumpfung....10
Parenchymverlust....11
Paßpunktbestimmung....69
Paßpunkte....68
Patienteninformationen....99
Peak....27, 31
peakAboveAverage....92
peakQuality....92
Pelvis....8
Perfusionsuntersuchung....19, 96
Periodizität....58
PG305....84
Phase....52
phase only matched filter....53
Phaseninformation....60
Phasenkodiergradient....18
Phasenkorrelation....53
Phasenspektrum....38
pixClassification....95
Pixel....31, 73
Pixeldaten....82
Pixelmenge....73, 98
Pixelraster....64
Pixelverschiebung....113
Pixelwiederholung....44
Plattformen....78
plattformunabhängig....23
Plattformunabhängigkeit....78
Plugin....77, 81, 86
PluginFilter....81
Plugin-Hauptfenster....113
Polarkoordinatendarstellung....61
Polygon-ROI....95, 111
Polygon-ROI-Schaltfäche....112
Polynom....45
Postscript....79
poststenotische Dilatation....14
Power-Kepstrum....54
Power-Spektrum....38
Präzession....18
prepareImages....92
Presets-Button....87
PresetsDialog....86, 87, 101
Primärharn....9
Programmiersprache....77
Programmierwerkzeug....78
Programmkomponenten....84
Projektgruppe....84
Projektion....27, 121
Properties....82
Proton....19
Psoas-Muskel....8
public....87
Pulssequenz....18
Pyelonephritiden....10

Q

Quelltext....78
Quelltexteditor....78
Quelltextkommentierung....79
Quelltextkonventionen....79

R

Randbedingungen....46
Rasterkoordinaten....44, 69
Rastertransformation....122
Rauschen....35
rauschunempfindlich....25
RealGrayImage....91
recall images....48
Rechengeschwindigkeit....91
Rechenzeitreduktionen....63
Rechnerkonfiguration....78
Referenzbild....24, 64
Referenzen....78
Registrierung....48
Reihenentwicklung....43, 55
Relaxation....20
Relaxationszeit....20
RENAL FUNCTION Analysis....23
RENAL_FUNCTION_Analysis....86
renale Funktionsanalyse....23, 29, 78, 82
renalen Funktionsanalyse....86
RenalFunctionAnalysis.properties....87
Renin....11
Renin-Angiotensin-Aldosteron-System....11
renovaskuläre Hypertonie....13
Repetitionszeit....20
Resampling....64
Resonanzfrequenz....18
ResultLog....91, 93
Richtungsgradientenbild....36
richtungsselektive Kantendetektoren....51
Robustheit....78
ROI....28, 83
ROI-Schaltflächen....121
Röntgen....17
Rotationen....25
Rotationsachse....17
Rücktransformation....41
run....81, 95

S

Säure-Base-Haushalt....7
savePresetsFile....88
ScaleDialog....103
scaledImage....100
scaler....101
Scanline....102
Schaltfläche....86
Schaltflächenleiste....111

Schichten ... 82
Schichtselektionsgradient ... 18
Schieberegler ... 23, 83
Schnittachse ... 67
Schnittbilddiagnostik ... 16
Schnittstellen ... 81
Schockniere ... 9
Schreibzugriff ... 93
Schrumpfniere ... 13
Schwellenwerte ... 75, 101, 118
Schwerpunkt ... 74
segementation ... 101
segmentationFinished ... 87
Segmentierung ... 25, 102
Segmentierungsalgorithmus ... 101
seitengetrennte Auswertung ... 64
sekundäre Hypertonie ... 12
Sequenz ... 24
setAcquisitionTime ... 88
setAverageGray ... 100
setCepstrumResult ... 88
setStdDeviation ... 100
setup ... 81
setXScale ... 100
setYScale ... 100
show ... 83
showMarks ... 99
showStandardDeviation ... 99
Signalamplitude ... 20
Signalanstieg ... 74
Signalintensität ... 23
Signal-Intensitäts-Kurven ... 27
Signal-Intensitäts-Zeit-Kurven ... 68, 72, 94, 124
signal-intensity-time-graphs ... 72
SIT ... 72, 86
SITAdjuster ... 94
SITAnalysis ... 94
SIT-Berechnung ... 111
sitFinished ... 87
SITResults ... 94
Skalarprodukt ... 43
Skalierungsfaktor ... 113
smoothCurves ... 99
smoothPolygon ... 103
Snakes ... 70
Sobel-Filterung ... 61
Sobel-Operator ... 36, 61
Sonderzeichen ... 79
Sonographie ... 16
Spiegelungstheorem ... 51
Spin ... 17
Spin-Echo-Techniken ... 19
Spin-Gitter-Relaxationszeit ... 20
Spin-Spin-Relaxationszeit ... 20
Spline-Kurven ... 44
Spracheigenschaften ... 78
stack ... 73, 81, 82
StackWindow ... 83
Standardabweichung ... 32, 73
start ... 95
startCepstrum ... 87
startSegmentation ... 87, 102
startSIT ... 87
Statusinformationen ... 111
Statuszeile ... 83
Steilheit ... 76
Stenose ... 13
Stenosegrad ... 12
Störsignal ... 35
Strahlenexposition ... 13, 15
Strahlverfolgungsverfahren ... 70
String ... 82
stückweise lineare Skalierung ... 33
Subtraktionsangiographie ... 14
Suchbereich ... 25
SWING ... 78
Symmetrieeigenschaften ... 63
symmetrische Phasenkorrelation ... 53
Syntaxhervorhebung ... 78
Systemkomponenten ... 83
Systemtheorie ... 54
Szintigraphie ... 15

T

T_1 20
T_2 20
T_2* ... 20
Tabellenkalkulations-Programm ... 29
Tabulatoren ... 29
template ... 69
template matching ... 48, 69
Tesla ... 15
Testphase ... 84
Textverarbeitung ... 79
Thermische Relaxationszeit ... 20
Thread ... 87
Tiefpaßfilter ... 35
Toleranzbereich ... 25, 61
Translationen ... 25
Translationskorrektur ... 26, 61, 66
trigonometrische Funktionen ... 55
Tubuli ... 9
Tubulopathie ... 10
Tubulusapparat ... 9
Tubulussystem ... 11
Tumoren ... 10
Turbo-Flash ... 15
Turbo-Flash-Aufnahmetechnik ... 82
Turbo-Flash-Sequenz ... 20, 23
Typisierung ... 78
Typumwandlung ... 87

Ü

Überladung ... 92
Überlagerungssatz ... 40
Überlappungsbereich ... 50

U

Ultraschall ... 16
Umsetzung ... 77
unlock ... 88
unscharfer Logik ... 70
Unter-ROI ... 97
Untersuchungsergebnis ... 27
Ureter ... 8
US ... 16
util ... 89

V

Vasokonstriktion ... 11
Vergrößerungs-Modus ... 112
Verkalkungen ... 19
Verschiebung ... 24

Verschiebungsinformation *52*
Verschiebungsvektor *65*
Vertauschungssatz *40*
Verzögerung *27*
Virtual Machine *78, 105*
Visualisierung *31*
Voreinstellungen *121*
Vorverarbeitung *61*
Vorverarbeitungsschritte *25*
vorzeichenbehaftet *81*
Voxel *31*

W

Wasserhaushalt *7*
Watershed-Algorithmus *70*
Weichteilkontrast *17*
Weiterverarbeitung *29*
Wendepunkt *27, 95*
Wendepunktbild *118*
Wertebereich *31*
WHO *12*
Wiener-Khintchine-Theorem *54*
WINDOWS95 ..... *105*
WINDOWS98 ..... *105*
WINDOWSNT 4.0 ..... *105*
Wirbelsäule *8, 64*
WORD97 ..... *79*
wrap around *58*

Z

Zeigerarithmetik *78*
Zeropadding *61*
Zielverzeichnis *99*
Zoom *83*
zweite Ableitung *27*
Zwerchfell *8*
Zystenniere *11*